OPTIMAL CONTROL OF QUEUEING SYSTEMS WITH NON-COLLABORATING SERVERS

A Thesis
Presented to
The Academic Faculty

by

Tuğçe Işık

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Industrial and Systems Engineering

Georgia Institute of Technology
August 2015

OPTIMAL CONTROL OF QUEUEING SYSTEMS WITH NON-COLLABORATING SERVERS

Approved by:

Professor Sigrún Andradóttir, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Hayriye Ayhan, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor J. G. "Jim" Dai
School of Operations Research and Information Engineering
Cornell University

Professor Robert D. Foley
School of Industrial and Systems Engineering
Georgia Institute of Technology

Professor Mark E. Lewis
School of Operations Research and Information Engineering
Cornell University

Date Approved: July 2015

To my parents, Nesrin and Mustafa Işık ...

ACKNOWLEDGEMENTS

I would like to thank my advisors Prof. Sigrun Andradóttir and Prof. Hayriye Ayhan for their guidance, patience and support throughout my doctoral studies. They have been the best role models and made immense contributions to my professional and personal development through the years.

I wish to also thank my committee members, Dr. Jim Dai, Dr. Robert D. Foley, and Dr. Mark E. Lewis, for their supportive suggestions and comments that has greatly improved this thesis.

I am grateful to Ms. Pam Morrison and Dr. Gary R. Parker for the their advice, knowledge and help that saved the day several times since my first year at ISyE. I am also grateful to Ms. Yvonne Smith and Mr. Mark Reese for for their support and kindness.

I am deeply thankful to have met my colleagues and friends at ISyE. Special shout out to İlke, Mallory, Haiyue, Bahar, Işıl, Oğuzhan, Diego, Ezgi, Gustavo, Rodolfo, Wajdi, Fanfang, Vinod, and Monica. They have made a very rocky road more than enjoyable. I also thank to my many other dear friends who shared my time at Georgia Tech.

Finally, I would like to send my sincerest gratitude to my family – my parents Nesrin and Mustafa, my dear sister Tuğba, and Bart for their endless support, love and encouragement that made the most difficult undertakings seem possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

This thesis is concerned with the effective management of a cross-trained workforce in manufacturing systems. In particular, we analyze non-collaborative queueing networks where multiple cross-trained (flexible) servers cannot work at a station together. We consider several related queueing networks with different objectives. Our contributions to the understanding of systems with non-collaborative flexible servers can be summarized in two parts. First, we characterize the structure of optimal server assignment policies and draw insights to improve decision making in these systems. Second, we develop easy-to-implement policies with near-optimal performance for systems where the optimal policy is difficult to implement or is not analytically tractable.

In the first study, our goal is to identify the server assignment policy that maximizes the long-run average throughput in tandem networks with finite buffers and non-collaborative flexible servers. For Markovian systems with two stations and two servers, we characterize the optimal server assignment policy and demonstrate that the structure of the optimal policy is insensitive to the service requirement distributions. For larger tandem networks, we propose server assignment heuristics that are near-optimal. Our numerical results suggest that in these systems, near-optimal throughputs can be achieved even if the server allocation decisions are made myopically. We also examine how lack of collaboration affects the performance of queueing systems with flexible servers. We show that the improvement that can be gained through collaboration is dependent on similarity of the tasks in the system, as well as the buffer sizes.

The second part focuses on tandem queueing networks with finite buffers and

non-collaborative flexible servers where server reassignments result in setup costs. For systems of arbitrary size with general service requirement distributions, we show that the policy that maximizes the long-run average profit becomes dedicated as the setup costs increase. We also characterize the profit-optimal server assignment policy for Markovian tandem lines with two stations, homogeneous tasks, and constant setup costs. Our results demonstrate that the structure of the optimal policy depends both on the magnitude of the setup costs and the buffer size, since dedicated server assignment policies become strictly suboptimal for any given setup cost as the buffer size increases. For systems with non-homogeneous tasks and/or non-constant setup costs, we provide near-optimal server assignment heuristics. Our computational results suggest that the relative performances of dynamic and dedicated server assignment policies are dependent on the structure of the tasks.

Finally, we extend our analysis to queueing networks with general topology and routing. For non-collaborative networks with infinite buffers, we formulate a linear program that yields an upper bound on the long-run average throughput. We also introduce a processor sharing scheme for general queueing networks, and identify the optimal processor sharing policy for tandem lines with infinite buffers and homogeneous tasks. For Markovian systems with two stations, finite buffers, and homogeneous tasks, we prove that processor sharing achieves the non-collaborative optimal throughput as the buffer size grows. To achieve near-optimal throughputs in systems where processor sharing is not implementable, we propose a class of round-robin server assignment policies and show that they approximate processor sharing in systems with two stations. We evaluate the performance of the proposed class of policies in systems with various topologies and finite buffers, and demonstrate that they are near-optimal.

CHAPTER I

INTRODUCTION

Facing high competition and rapid market changes, manufacturers look for ways to increase agility in the production process to remain successful. Widely used strategies include designing flexible production processes and factory layouts, or improving material handling. Although these strategies greatly contribute to the responsiveness of the manufacturing system when applied successfully, their implementation may be impossible or very costly for some manufacturers. Under these circumstances, workforce agility gains further importance. As a result, cross-training has become a popular tool to facilitate immediate response to the system needs beyond utilizing process flexibility or routing flexibility.

It is known that various performance measures, including system costs and throughput, can be improved when servers are cross-trained (flexible) [40]. However, the existing literature on cross-training does not distinguish the benefits of collaboration from the benefits of dynamic server allocation, which are both potential advantages of server flexibility. Furthermore, the majority of the earlier studies consider systems with ample space and equipment [61, 66] where multiple servers can work at a workstation at the same time. Thus, the resulting server allocation strategies for systems with cross-trained servers often make use of collaboration.

In a manufacturing system with multiple workers and tasks, there can be restrictions that will render collaboration of the workers impossible. These restrictions can be caused by the nature of the jobs, physical limitations in the workspace, and the lack of sufficient tooling. In flexible manufacturing systems, the amount of available tooling is a restrictive factor, along with machine and labor constraints, and limits

the production capacity [28, 33, 43]. In production facilities that use molds, dies, or other specialized fixtures, acquisition of new tools can be as costly as the manufacturing machinery itself [25]. These restrictions often prevent the implementation of collaborative worker assignment policies.

This thesis aims to investigate the benefits of having flexible servers in manufacturing systems when the servers cannot collaborate, and is concerned with finding effective server assignment policies for such systems. Our analysis demonstrates that cross-training can improve the performance in non-collaborative systems, if the workforce is managed effectively.

We start our analysis by studying a tandem manufacturing system with finite buffers and flexible servers. The servers can be reassigned to tasks without any setups, but are unable to collaborate at a station. For tandem lines with general service requirements, we prove structural results on the server assignment policy that maximizes the long-run average throughput. We characterize the throughput-optimal server assignment policy for the Markovian tandem network with two stations. We also provide numerical results that indicate that the threshold structure that is optimal in Markovian systems is insensitive to the service requirement distributions. Furthermore, we develop heuristic policies that achieve near-optimal throughputs for tandem lines with more than two stations. Our numerical results suggest that the structure of the optimal server assignment policy for larger tandem networks remains of threshold type, and dynamic allocation of flexible workforce results in significant performance improvement even if the system is non-collaborative. To provide further insights on the benefits of server flexibility in tandem networks, we compare collaborative and non-collaborative systems. We show that the loss in the long-run average throughput in non-collaborative systems is mitigated by the similarity of the tasks and large buffer sizes. Thus, dynamic server allocation can compensate for the lack of collaboration when the tasks are homogeneous.

In manufacturing systems, reallocating workers to tasks may require setups that lead to losses in productivity [41, 53]. When the setups are significant, cross-training might not be as beneficial since frequent movement of the servers among tasks is detrimental to system performance. To evaluate this trade-off, we study a non-collaborative tandem network where each server reassignment results in setup costs. We show that regardless of system size and service requirement distributions, the long-run average profit is maximized by a dedicated server assignment if the setup costs are large enough. We analyze a Markovian two-stage tandem network with homogeneous tasks and constant setup costs, and completely characterize the profit-optimal server assignment policy. Our results show that the optimal policy is of double-threshold type, and the optimal thresholds are dependent on the magnitude of the setup costs. We also show that cross-training becomes advantageous regardless of setup costs if the buffer size is large enough. For systems with non-homogeneous tasks and/or setup costs that are dependent on the location of the servers, we develop server assignment heuristics. We demonstrate through numerical experiments that dedicated server assignments can be near-optimal in systems with non-homogeneous tasks if the buffer size is small, while they are outperformed by the dynamic server assignment heuristics in systems with homogeneous tasks at all buffer sizes.

Although our analysis of tandem manufacturing systems provides valuable insights on the benefits of cross-training in non-collaborative settings, flexible manufacturing systems often operate with more general routing and multiple job types. In the last study of this dissertation, we aim to develop non-collaborative server assignment strategies applicable in general queueing networks with finite buffers. We first consider a non-collaborative queueing network with infinite buffers and develop a linear program that provides an upper bound on the achievable long-run average throughput. We then interpret the optimal solution of the linear program as a processor

sharing scheme we developed for queueing networks. We determine the optimal processor sharing policy for tandem networks with arbitrary number of stations and homogeneous tasks, as well as two-stage tandem lines with non-homogeneous tasks. For networks where processor sharing is not applicable, we develop timed round-robin server assignment policies. Moreover, we show that the throughput of these policies converge to that of processor sharing in systems with two stations and homogeneous tasks as the server reassignments become more frequent. For larger queueing networks with finite buffers and tandem or non-tandem structures, we evaluate the performance of the round-robin policies via a simulation study. Our results indicate that the round-robin policies yield near-optimal throughputs in networks with finite buffers.

The rest of this dissertation is organized as follows. In Chapter 2, we provide an overview of the literature on queueing systems with flexible servers. In Chapter 3, we present our results on tandem networks with non-collaborative flexible servers. In Chapter 4, we study non-collaborative systems with setup costs. In Chapter 5, we develop server assignment policies for non-collaborative networks with general routing. In Chapter 6, we discuss our conclusions. Finally, in Chapter 7, we outline future research directions. Additional numerical results for Chapter 4 can be found in Appendices.

CHAPTER II

LITERATURE REVIEW

As agile manufacturing systems grow in popularity, systems with cross-trained workforce have attracted vast interest among both practitioners and researchers. Consequently, there is extensive literature on the design and control of queueing systems with flexible servers. In this chapter, we present related literature on the dynamic allocation of flexible servers, as well as on different flexibility configurations and collaboration frameworks. Section 2.1 focuses on the previous work that considers design issues in systems with flexible servers. In Sections 2.2 and 2.3, we provide a review of the literature on dynamic server allocation in systems with or without operational costs, respectively. Section 2.4, presents an overview of the existing work on systems with finite buffers. In Section 2.5, we review the literature on processor sharing. Finally, Section 2.6 summarizes our contributions to the literature.

2.1 Design of Systems with Flexible Servers

There are numerous research papers which focus on benefits of workforce flexibility, skill chaining, and cross-training. A number of early papers seek to gain managerial insights by investigating ways to increase agility in the production system by the efficient use of workforce and process flexibility, as well as examining the benefits of agile workforce as opposed to its costs and managerial challenges. Bartholdi and Eisenstein [19] study an m-station production system with n servers that can move among those stations. They show that if the servers are sequenced from fastest to slowest on the production line, by following a simple predefined rule, the production rate will converge to its maximum and stable partition of work will emerge. Jordan and Graves [42] develop principles on the benefits of process flexibility. They show that limited

flexibility can yield the benefits of total flexibility if configured to chain products or plants. Sheikzadeh et al. [63] compare full machine flexibility and chaining that limits number of the job types that a server can process to two. They demonstrate that the benefits which can be gained by increasing the chain size are diminishing and most of the benefits of total flexibility can be captured by chaining. In a more recent paper, Gurumurthi and Benjaafar [35] study a queueing system with multiple customer classes and heterogeneous servers where the customers have the flexibility of being processed by different servers. They provide an analytical model and examine the relationship between flexibility and system throughput. Andradóttir et al. [10] develops design principles for systems with flexible servers and identify desirable characteristics of flexibility structures. They evaluate commonly studied flexibility structures in terms of these criteria.

There are also studies on increasing benefits of workforce flexibility by improving other elements in the manufacturing environment. Bischak [22] proposes a U-shaped manufacturing module with fewer workers than work stations in which workers are allowed to move between the stations of the module. Throughput of this system is compared to the throughput of a serial manufacturing line with one dedicated server at each station. The simulation results suggest that moving worker modules can provide flexible capacity without the use of buffers. Agrihothri et al. [1] find the optimal mix of dedicated and flexible servers to minimize average costs in a system of parallel queues with a given total number of servers. They run simulations to investigate the impact of various system parameters on the server mix that minimizes the total average service costs.

An important advantage of having flexible workforce is that servers are allowed to move among stations; therefore dynamic server allocation in response to the system state is possible. There is extensive literature on dynamic decision making in manufacturing systems with flexible workforce in addition to the papers that are concerned

with finding the most beneficial configurations of flexibility. Hopp and Van Oyen [40] give a classification of different collaboration structures for flexible servers and provide an extensive summary of the related literature. We provide an overview of the previous work on dynamic server allocation in the remaining sections of this chapter.

2.2 Maximizing Throughput with Flexible Servers

The majority of existing work on dynamic allocation of flexible servers is focused on the collaborative settings in which either multiple servers can work on a single job or multiple jobs can be processed simultaneously at a station. Several performance measures including long-run average throughput and costs were considered for queueing networks with various configurations.

A number of papers focus on maximizing the long-run average throughput in tandem queueing networks with finite buffers and no arrivals. Among these, Andradóttir et al. [11] describe a simple server assignment policy with primary assignments and contingency plans that yields near-optimal throughput for N station-N server tandem systems with $N \geq 2$, and characterize the optimal policy for two-station, two-server case. Andradóttir and Ayhan [9] look at the same problem with two stations and characterize the optimal policy for $M = 3$ servers. Hasenbein and Kim [37] study the same system and prove the structure of the optimal policy conjectured in [9] for two stations and M servers. Andradóttir et al. [13] study a tandem queueing system with both dedicated and flexible servers, and compare the improvement obtained by adding a flexible server to the improvement obtained by adding a buffer space or server to the system. They show that having only one flexible server is sufficient for achieving near-optimal throughput in comparison to systems where all servers are flexible. Andradóttir et al. [7] consider tandem lines with heterogeneous servers who are synergistic, i.e, they work more effectively when in teams compared to when they are on their own. They characterize the optimal policy for the case with two stations

and two servers, and also provide sufficient conditions for the optimal policy to utilize the synergy between servers at all times in larger networks. Finally, Andradóttir et al. [8] find the optimal dynamic server assignment policy when servers are inefficient when collaborating, i.e. they are faster when they work on their own.

Several other papers focus on systems that are not tandem, that have infinite buffers, or that have external arrivals with the objective of maximizing the average long-run throughput. Here we present a selection of the previous work. Andradóttir et al. [12] study a queueing network with infinite buffers and collaborative flexible servers. They compute upper bounds on the maximal capacity using fluid models and construct generalized round-robin policies that guarantee that the capacity will be arbitrarily close to the computed upper bounds. Tekin et al. [64] did a fluid limit analysis of a system with M servers and K customer classes where servers work in parallel. Their objective is to maximize system throughput when the demand may exceed the capacity for service. They formulate linear programming problems to compute several desired quantities and develop generalized round-robin policies that achieve the desired throughput as long as it is feasible. Andradóttir et al. [6] analyze a queueing network with flexible servers where servers are subject to failure and the routing is probabilistic. They show that the maximal capacity is tightly bounded by the solution of a linear programming problem. Moreover, they use the solution of the linear programming problem to construct generalized round-robin policies. Ahn and Righter [5] analyze systems of n stations in tandem with infinite buffers between the stations when servers are trained to work on a subset of consecutive stations. In several settings, the optimal policy is shown to be either LBFS (last buffer first-served) or FBFS (first buffer first-served).

Argon and Tsai [65] study a system with instantaneous assembly and disassembly operations that has two servers and two intermediate feeder stations that are in parallel, and characterize the optimal server assignment policy. They also develop

heuristic policies for systems with three or more feeder stations. Arumugam et al. [17] provide structural results about the server assignment policy that maximizes the throughput for a two station system with external arrivals when servers can collaborate on the same job. They also provide a numerical analysis for the case where the servers cannot collaborate on a single job but can work at the same station. Hopp et al. [39] evaluate two cross-training architectures, namely capacity balancing and overlapping zones, in terms of maximizing system throughput. They develop heuristics for these architectures and give the corresponding Markov Decision Process formulations. Van Oyen et al. [66] analyze a serial system with a general arrival process and no blocking, assuming servers are completely flexible and collaborative. They prove that the expedite policy, which assigns all servers to one job at a time, minimizes the work in progress and maximizes throughput along every sample path.

2.3 Systems with Operational Costs

Operational costs in a manufacturing system are also affected by cross-training and dynamic server allocation. Systems with cross-trained workforce have further ability to balance the workload to reduce inventory costs, as well as new operational challenges due to mobility of the servers. As a result, there are several studies that focus on systems with flexible servers and setup or holding costs.

Many of the papers mentioned in the previous paragraph consider clearing systems with no arrivals. In particular, Glazebrook [34] studies the stochastic scheduling problem for a single-server clearing system with setup costs where the jobs are processed following a precedence relationship, but the server is allowed to switch between jobs. Ahn et al. [3] argue the benefits of having cross trained workers in order to minimize holding costs. They consider a tandem clearing system with two stations and two identical, cross-trained servers. In the presence of holding costs, they characterize the conditions under which it is optimal to assign both servers to the upstream or

downstream stations. Schiefermayer and Weichbold [60] minimize the holding costs for a two-stage clearing tandem queue with collaborative identical servers that can work at either station.

Some other papers look at systems with both flexible and dedicated servers. A clearing system of two parallel queues with one dedicated and one flexible server is studied by Ahn et al. [4] and the policy that minimizes the holding costs is completely characterized. A two-stage clearing system with no external arrivals, dedicated servers at each station, and one additional server that can be dynamically allocated is considered by Farrar [31]. For linear holding costs, it is shown that the optimal server allocation policy never increases the effort devoted to a station after a service completion at that station (i.e., optimal policy is transition monotone). A similar clearing system of two tandem queues with dedicated servers at each stage and additional flexible resources is considered by Wu et al. [67]. For systems with or without machine failures, they show that a transition monotone policy that minimizes the holding costs exists.

There is also extensive literature on systems with outside arrivals and operational costs. For example, Hajek [36] proves the existence of optimal polices that are described by switching curves in the two-dimensional state space of systems with two service stations and linear costs. Mendelbaum and Stolyar [51] study a parallel queueing system with flexible servers in heavy traffic and show that a generalized $c\mu$ rule minimizes both instantaneous and cumulative costs asymptotically.

Several papers study systems with outside arrivals and holding costs. Pandelis [54] shows the optimality of transition monotone policies for two-stage tandem queueing systems with linear holding costs and with or without arrivals when there is a dedicated server at each station together with N additional flexible servers. Reiman and Wein [57] consider a system with external arrivals, general service time distributions, and holding costs that serves two customer classes with a single flexible server. They

study the system with setup costs and setup times, and solve the diffusion control problem corresponding to the setup problem. A tandem queueing system with a single server, infinite buffers, and Poisson arrivals is studied by Duenyas et al. [29]. The optimal policy is partially characterized and a simple heuristic scheduling policy is proposed.

Kırkızlar et al. [44] study the dynamic server allocation problem in tandem lines with holding costs. They characterize the policy that maximizes the long-run average profit for systems with arbitrary number of stations when the server rates are structured and for two-station systems with general service rates when the costs are linear. Rosberg et al. [59] also study two M/M/1 stations in tandem where the service rate at station 1 is chosen as a function of system state. They characterize the optimal policy with a switching function in the case that the holding cost is a linear function of the number of jobs in the buffers. Iravani et al. [41] consider a two-stage tandem queue with a single moving server, Poisson arrivals, and general service times. They show that the policy that minimizes the holding costs is greedy in the second stage and develop a threshold-type heuristic for the first stage. Ahn et al. [2] study a two-stage tandem queue with holding costs under different collaboration schemes and characterize the optimal policy under simple conditions. A two-station serial system with infinite buffers and Poisson arrivals that has switching and holding costs is studied by Mayorga et al. [52]. Sennott et al. [61] consider a K-station tandem line with setup times, setup costs, and holding costs, where each station has its own dedicated server and there is one fully flexible floating server. They develop bounds for the minimum long-run average cost and give a numerical analysis that shows the benefits of having a flexible server. Argon and Andradóttir [16] analyze the effects of pooling for single server queues with cross-trained servers and identify sufficient conditions under which pooling reduces the departure time and improves the holding costs.

Finally, there are papers that consider tandem systems with setup costs and outside arrivals. Batta et al. [20] study a system with multi-type customers and flexible servers that can serve different customer classes. They develop heuristics with the objective of minimizing the total staffing and setup costs subject to a service level constraint. Koole [48] studies the assignment of a single server to two queues with Poisson arrivals to minimize holding and setup costs, and develops a threshold-type policy. Similarly, Duenyas and van Oyen [30] analyze a system of queues with Poisson arrivals and a single server with the objective of minimizing holding and setup costs. They partially characterize the optimal policy and provide a simple scheduling policy. Arumugam et al. [52] study a tandem queueing system with switching costs and show that the optimal policy follows a complex state-dependent structure even in the two station case. They propose a simpler heuristic policy that involves moving only one server. Andradóttir et al. [14] study a two-station tandem system with collaborative servers and setup costs. They characterize the optimal policy for small buffer sizes assuming the tasks or servers are homogeneous, and give a partial result for systems with no buffer space and general service rates.

2.4 Systems with Finite Buffers

In manufacturing systems, there is often limited capacity for work-in-process inventory. However, due to the curse of dimensionality, the analysis of the finite-buffered systems with large number of stations is challenging even under limiting assumptions on routing and topology. Most of the previous work on dynamic server allocation in systems with flexible servers and finite buffers focuses on tandem lines [9, 11, 14, 17, 37, 44].

On the other hand, most of the work on finite-buffered systems with non-tandem configurations focuses on approximate analysis of these systems or systems with no exogenous arrivals, and are interested in descriptive analysis rather than control of

these systems. For example, Shanthikumar and Yao [62] study the server allocation problem in a closed queueing network with finite buffer capacities, and show that the throughput is concave in the number of servers at a station. Zhuang and Hindi [68] consider a similar closed queueing network with finite buffers and exponential service times. They develop a decomposition method for systems with state dependent routing probabilities. Smith [50] proposes a decomposition algorithm for finite buffered closed queueing networks with cycle, merge, and split topologies. Pestien and Ramakrishnan [55] compute the expected number of busy servers in a closed, cyclic, and discrete time queueing network. They also estimate the asymptotic cycle time for a single job in a similar system with no buffer spaces. A queueing network with exogenous arrivals and finite buffers is studied by Smith [49], and provides estimates for throughput, work-in-progress, and sojourn times. These estimates were evaluated for networks with series, merge, or split topologies. Note that none of these papers consider dynamic allocation of servers.

The only papers we are aware of that study server allocation in non-tandem queueing networks with finite buffers are by Tsai and Argon [15, 65], who study assembly-type Markovian systems with or without operational costs. For systems without operational costs, they characterize the server assignment policy that maximizes the long-run average throughput for systems with two intermediate feeder stations and instantaneous disassembly and assembly operations [65]. For a single-server assembly queue with holding and setup costs, they partially characterize the server assignment policy that minimizes long-run average costs and provide sufficient conditions under which working on one order at a time is optimal [15].

2.5 Shared Flexible Servers

When the tasks are suitable, flexible servers can be shared by multiple jobs that are simultaneously served. A processor sharing mechanism can also be utilized as an idealized model for manufacturing and service systems where the servers attend multiple jobs. In particular, processor sharing has been studied as a limiting model for time-shared computing facilities or communication networks with shared bandwidth. Here we present a selection of related work rather than providing an extensive treatment of the literature.

The seminal paper by Kleinrock [45] studies a a single-server time-shared queue with a service rate of μ, where each job in the queue receives service in round-robin fashion, and analyzes the limiting processor sharing queue where each job in the queue receives infinitesimal service infinitely often. Furthermore, it is shown that the processor sharing queue is equivalent to a system where each job is processed at a rate of $\frac{\mu}{n}$ when there are n jobs in the system. Several other papers consider equivalence relations between time-shared queues and other queueing mechanisms. Coffman and Kleinrock [27] compare average time spent in the system for a single round-robin queue with a system of N feedback queues where customers at the lowest indexed queue receive a quantum of service and move to the next queue. They also examine the limiting processor sharing case and provide comparisons with the first-come-first-served and shortest-job-first service disciplines. Resing et al. [58] study the sojourn times in an M/M/1 feedback queue and show that they converge to the sojourn times in an M/G/1 processor sharing queue. Borst et al. [23] study the delay distribution in a G/M/1 queue, and show an equivalence between processor sharing and service in random order.

Another group of papers are focused on the analysis of waiting or sojourn times in processor sharing queues. For example, optimal scheduling algorithms that minimize a cost function that depends on the waiting and sojourn times for time-shared

systems are studied by Kleinrock and Nilsson [47]. Similarly, Kleinrock and Muntz [46] consider the scheduling algorithms for time-shared systems and study the average response time using the simpler processor sharing limit. Fayolle et al. [32] look at a single server processor sharing system with M job classes and study the mean response time conditioned on the service requirement. Brandt and Brandt [24] consider Markovian systems with state dependent processor sharing and study the conditional wait time of a job given the required service time. Ayesta et al. [18] consider an M/G/1 queue where a server goes on a vacation whenever the queue becomes empty and they determine the sojourn time distribution. Chen and Jordan [26] consider throughput as a performance measure rather than the soujourn times. They study an M/M/1 processor sharing queue and develop closed-form expressions for the average service rates observed by the jobs and the queue.

2.6 Contributions

Our work extends the existing literature in several ways. All the papers we have encountered so far analyze dynamic allocation of flexible servers assuming some form of collaboration is possible. Most of the papers in the literature assume that when the servers are collaborative, two or more servers can work on the same job. Other papers, such as Ahn et al. [4], study systems in which servers are not allowed to work on the same job, but can work on different jobs from the same queue. Some others, such as Ahn et al. [2] and Arumugam et al. [17], consider both collaboration schemes. By contrast, we identify optimal or near optimal server assignment policies for a non-collaborative setting which is better suited for manufacturing systems with tooling constraints [28, 33, 43]. We also address the distinction between the benefits of dynamic server allocation and collaboration in systems with flexible servers.

Furthermore, tandem systems with setup costs and outside arrivals are not widely studied in the literature. The few papers that consider such systems present partial

results due to the higher dimensionality of the state space and the complex structure of the optimal policies. However, previous work on allocation of flexible servers in the presence of setup costs considers either single-server systems or assumes servers are collaborative. To our knowledge, our work in Chapter 4 is the first to (i) consider a non-collaborative system and (ii) fully characterize the policy in a setting of interest.

We also propose a novel processor sharing scheme since the previous work considers processor sharing at a single queue rather than a network of queues. Furthermore, using our results on processor sharing, we are able to develop dynamic server allocation policies for non-collaborative queueing networks with general topology and routing. To the best of our knowledge, our work in Chapter 5 is the first to model processor sharing queueing networks and to consider dynamic server allocation in non-tandem queueing networks with general routing and finite buffers.

CHAPTER III

OPTIMAL CONTROL OF NON-COLLABORATIVE TANDEM NETWORKS

3.1 Introduction

In this chapter, we study a non-collaborative tandem production line with cross-trained workers. Our aim is to determine the advantages of cross-training and dynamic server allocation in such systems, as opposed to collaboration, and identify the dynamic server allocation policy that maximizes the long-run average throughput.

We consider a system of N tandem queues, finite buffers between the stations, and N flexible servers. At any time, at most one job can be processed at each station and a server can work on at most one job. The system works under manufacturing blocking. We assume an infinite amount of raw material is ready to be processed at the first station and there is infinite space after the last station. Thus, a new job enters the system whenever station 1 is empty and the last station is never blocked. The service requirements at a station are independent and identically distributed random variables that are independent of the service requirements at other stations. Also, without loss of generality, the mean service requirement is one at all stations. On a sample path, a service requirement is realized for each job and the station the job is at following some distribution. The service requirement at a station is then depleted at the rate of the server assigned to that station. Let $0 \leq B_j < \infty$, $j \in \{1, 2, \ldots, N-1\}$, denote the size of the buffer between stations j and $j+1$, and μ_{ij}, $i, j \in \{1, 2, \ldots, N\}$, denote the processing rate of server i at station j. The system is restricted to be non-collaborative, therefore multiple servers are not allowed to work together at a station. However, servers are allowed to switch among stations and it is assumed that travel

times between stations are negligible.

The remainder of the chapter is organized as follows. In Section 3.2, we present results for systems with N stations and structured service rates. In Section 3.3, we formulate a Markov Decision Process corresponding to the N-station system with arbitrary service rates. Then we characterize the optimal server assignment policy for the two-station Markovian system and evaluate the sensitivity of the optimal policy to the service requirement distribution in Section 3.4. In Section 3.5, we provide a comparison of collaborative and non-collaborative Markovian systems. In Section 3.6, we develop heuristic policies and evaluate their performance for non-collaborative systems with three or more stations where the service rates depend only on the server. Section 3.7 summarizes our conclusions.

3.2 Structural Results for N-Server Systems

In general, finding the optimal server assignment policy for the N-station, N-server system is complicated due to the high-dimensional state space and large number of available actions. The following proposition shows we can restrict our attention to the non-idling policies as an idling policy cannot be strictly optimal.

Proposition 1. *Proposition For an N-station tandem line with N non-collaborative servers and service requirements with general distributions, there exists a non-idling optimal server assignment policy.*

Proof. Let π be a non-collaborative idling policy. Any time the policy π idles the servers $\{i_1, i_2, \ldots, i_k\}$, there are stations $\{j_1, j_2, \ldots, j_k\}$ that are unattended, where k is the number of idled servers. Let π' be the non-idling policy that is identical to the policy π except for that it assigns the idled servers $\{i_1, i_2, \ldots, i_k\}$ to the unattended stations $\{j_1, j_2, \ldots, j_k\}$ whenever they are idled under the policy π.

Let $\mu^{\pi}_{j,n}$ be defined as the average service rate observed by the n^{th} job at station j under policy π. Then, we have $\mu^{\pi}_{j,n} \leq \mu^{\pi'}_{j,n}$ for all $j \in \{1, \ldots, N\}$ and for all n, by the

way the policy π' is constructed, and the sojourn times under policy π are at least as large as under policy π'. Thus, for any idling policy π, we can construct a non-idling policy π' that is at least as good as π, and the result follows. □

The number of non-idling actions available in a particular system state can be as high as $N!$ (when there are no stations blocked or starved). We focus on a special case and identify the optimal server assignment policy for a system where for each station there exists one distinct server that is better than all other servers at that station.

Theorem 1. *Consider an N-station tandem line with N non-collaborative servers and service requirements with general distributions. Suppose there exist $i_1, \ldots, i_N$ with $\{i_1, i_2, \ldots, i_N\} = \{1, 2, \ldots, N\}$ such that $\mu_{i_j j} = \max_{1 \le i \le N} \mu_{ij}$ for all $j \in \{1, 2, \ldots, N\}$. Then the policy that assigns server i_j to station j at all times for all $j \in \{1, 2, \ldots, N\}$ is optimal.*

Proof. Let us assume, without loss of generality, that we have $i_j = j$ for $j \in \{1, 2, \ldots, N\}$, and let π^* be the policy that keeps server j at station j at all times. We prove that the policy π^* is optimal.

Let $t^{\pi}_{j,n}$ denote the n^{th} departure time from station j under a non-idling policy π, and $\phi_{j,n}$ denote the service requirement of the n^{th} job at station j for $n \in \{1, 2, \ldots\}$, $j \in \{1, 2, \ldots, N\}$. Also let $\mu^{\pi}_{j,n}$ denote the average service rate that the n^{th} job observes at station j under policy π, so that $\frac{\phi_{j,n}}{\mu^{\pi}_{j,n}}$ is the time spent at the j^{th} station by the n^{th} job. More specifically, let $l^{\pi}_{j,n}$ be the number of server reassignments that occur during the time the n^{th} job spends at station j when the policy π is employed. Also let $\phi^{\pi}_{j,n}(q), q \in \{1, \ldots, l^{\pi}_{j,n} + 1\}$, denote the service requirement fulfilled between the $(q-1)^{th}$ and q^{th} server reassignments and let $\mu^{\pi}_{j,n}(q), q \in \{1, \ldots, l^{\pi}_{j,n} + 1\}$, denote the rate of the server assigned to station j for the time between the $(q-1)^{th}$ and q^{th} server reassignments. Then, with the convention that $\frac{0}{0} = 0$, $\phi_{j,n} = \sum_{q=1}^{l^{\pi}_{j,n}+1} \phi^{\pi}_{j,n}(q)$

and $\frac{\phi_{j,n}}{\mu^{\pi}_{j,n}} = \frac{\phi^{\pi}_{j,n}(1)}{\mu^{\pi}_{j,n}(1)} + \cdots + \frac{\phi^{\pi}_{j,n}(l^{\pi}_{j,n}+1)}{\mu^{\pi}_{j,n}(l^{\pi}_{j,n}+1)}$. Furthermore, for $t^{\pi}_{j,n}$, $j \in \{1,2,\ldots,N\}$, $n \geq 1$, the following recursion holds with the convention that $t^{\pi}_{j,n} = 0$ if $j \in \{0, N+1\}$ or $n \leq 0$:

$$t^{\pi}_{j,n} = \max\left\{t^{\pi}_{j-1,n} + \frac{\phi_{j,n}}{\mu^{\pi}_{j,n}}, t^{\pi}_{j,n-1} + \frac{\phi_{j,n}}{\mu^{\pi}_{j,n}}, t^{\pi}_{j+1,n-B_j-1}\right\}. \tag{1}$$

Note that equation (1) holds because the n^{th} job starts service at station j either upon arrival ($t^{\pi}_{j,n} = t^{\pi}_{j-1,n} + \frac{\phi_{j,n}}{\mu^{\pi}_{j,n}}$) or after the $(n-1)^{th}$ job leaves station j ($t^{\pi}_{j,n} = t^{\pi}_{j,n-1} + \frac{\phi_{j,n}}{\mu^{\pi}_{j,n}}$), and might be blocked at station j upon finishing service ($t^{\pi}_{j,n} = t^{\pi}_{j+1,n-B_j-1}$).

Assume that there exists a job n and station a j such that $t^{\pi}_{j,n} < t^{\pi^*}_{j,n}$. Let job n' be the first such job and let j' be the station with the smallest index such that $t^{\pi}_{j,n'} < t^{\pi^*}_{j,n'}$. Then, we have $t^{\pi}_{j,n} \geq t^{\pi^*}_{j,n}$ for $n = n'$, $j < j'$ and for all $n < n'$, $j \in \{1,\ldots,N\}$. Furthermore, $\mu^{\pi}_{j,n} \leq \mu^{\pi^*}_{j,n} = \mu_{jj}$ for all $j \in \{1,\ldots,N\}$, $n \geq 1$ by our assumption that $\mu_{jj} = \max_{1 \leq i \leq N} \mu_{ij}$. Thus, $t^{\pi}_{j'-1,n'} + \frac{\phi_{j',n'}}{\mu^{\pi}_{j',n'}} \geq t^{\pi^*}_{j'-1,n'} + \frac{\phi_{j',n'}}{\mu^{\pi^*}_{j',n'}}$, $t^{\pi}_{j',n'-1} + \frac{\phi_{j',n'}}{\mu^{\pi}_{j',n'}} \geq t^{\pi^*}_{j',n'-1} + \frac{\phi_{j',n'}}{\mu^{\pi^*}_{j',n'}}$, and $t^{\pi}_{j'+1,n'-B_{j'}-1} \geq t^{\pi^*}_{j'+1,n'-B_{j'}-1}$, contradicting $t^{\pi}_{j',n'} < t^{\pi^*}_{j',n'}$. It follows that $t^{\pi}_{j,n} \geq t^{\pi^*}_{j,n}$ for all $j \in \{1,\ldots,N\}$, $n \geq 1$ and the policy π^* is optimal. □

Remark 1. *Theorem 1 shows that when each server works faster at a distinct station, the optimal policy is to dedicate the servers where they are the fastest. Thus, there is no benefit from cross-training the servers.*

Note that the result given in Theorem (1) is quite intuitive, since when each server is specialized at one of the tasks switching servers decreases the service rates at all of the stations to which servers are reassigned. The following proposition shows a similar result when a subset of the servers are specialized.

Proposition 2. *In a tandem line with N stations and N servers, if there is a set of servers $I \subset \{1,2,\ldots,N\}$ and a set of stations $J = \{j_i : i \in I\} \subset \{1,2,\ldots,N\}$ such that $\mu_{ij_i} \geq \mu_{kj_i}$ for all $k \neq i$, $i \in I$ and $\mu_{ij} \leq \mu_{kj}$ for all $i \in I$, $j \notin J$, $k \notin I$ then there exists an optimal policy that keeps server $i \in I$ at station j_i at all times for all $i \in I$.*

Proof. Let I' be a subset of the set I, and suppose there exists an optimal policy π that assigns each server $i \in I' \subset I$ to a station j such that $j \neq j_i$ for an amount of time.

We define the set of stations $J' = J'_1 \cup J'_2 \cup J'_3$, where J'_1 is the set of stations $j \in J$ to which a server $i \notin I$ is assigned, J'_2 is the set of stations $j \in J$ to which a server $i \in I$ with $j \neq j_i$ is assigned, and J'_3 is the set of stations $j \notin J$ to which a server $i \in I$ is assigned. Note that for each station in the set J'_1, there must be a server $i \in I$ assigned to a station $j \notin J$, and for each station in the set J'_3, there must be a server $i \notin I$ assigned to a station $j \in J$. Therefore, $|J'_1| = |J'_3|$. Furthermore, for each $j \in J'_2$ the server i with $j = j_i$ must be assigned to a station $j' \in J'_2 \cup J'_3$.

We can construct a policy that assigns every server $i \in I$ to the corresponding station j_i and is at least as good as the policy π as follows. Consider a policy π' that is identical to policy π except when policy π moves the servers $i \in I'$ away from stations $j_i \in J$. Instead, the policy π' assigns the servers $i \notin I$ that are assigned to the stations $j \in J'_1$ by policy π to the stations $j \in J'_3$, and assigns the servers $i \in I$ that are assigned to the stations $j \in J'_3 \cup J'_2$ by policy π to the stations $j_i \in J'_1 \cup J'_2$. Note that under the policy π', the service rates at the stations $j \in J'$ are always at least as high as under the policy π, thus the policy π' performs at least as well as the policy π, finishing the proof. □

3.3 Problem Formulation

We give a formulation of the problem with the assumption that the service requirements are exponentially distributed. For the N-station system, we define the stochastic process $\{X^\pi(t) : t \geq 0\}$ for a policy $\pi \in \Pi$, where Π denotes the set of all possible server assignment policies, as follows: $X^\pi(t) = (s_1, \ldots, s_{N-1})$, $s_i \in \{0, \ldots, B_i+2\}$, $\forall i$, where s_i is the number of jobs that have been processed at stations $\{1, \ldots, i\}$ but not at stations $\{i+1, \ldots, N\}$. Let us call the set of all possible states S.

From now on, we assume that Π is the set of all Markovian stationary deterministic server assignment policies corresponding to the state space S. Note that the stochastic process $\{X^{\pi}(t)\},\ \pi \in \Pi$, is a continuous time Markov chain. There exists a finite uniformization constant $q \leq \sum_i \max_j \mu_{ij}$. Therefore the chain is uniformizable, and the continuous time optimization problem can be translated into an equivalent discrete time Markov decision problem. In particular, Andradóttir et al. [11] show that maximizing the steady-state throughput of the original system is equivalent to maximizing the steady-state departure rate for the embedded discrete time Markov chain.

We let $a_{\sigma_1\sigma_2\ldots\sigma_N}$ denote an action, where $\sigma_i \in \{0, 1, \ldots,\ N\}$ shows the allocation of server i, with the convention that if $\sigma_i = 0$, then server i is idled, and if $\sigma_i = j > 0$, then server i is assigned to station j. Also let $A_s = \{a_{\sigma_1\sigma_2\ldots\sigma_N} : \sigma_i \in \{0, 1, \ldots, N\}, \sigma_i \neq \sigma_k \text{ for } i \neq k \text{ with } \sigma_i\sigma_k > 0\}$ denote the set of allowable actions in state $s \in S$.

Lastly, we let μ_{ij}^k for $i, j \in \{1, \ldots, N\}$ and $k \in \mathbb{N}$ denote the k^{th} power of the service rate μ_{ij} throughout the chapter, unless stated otherwise.

3.4 Optimal Policy for Two-Server Systems

In this section, we consider a two-station system, and completely characterize the optimal policy when the system is Markovian. There are two cases to be examined: Either there is no dominating server (i.e., each server is faster at one of the stations) or there is a dominating server (i.e., the same server is faster at both stations). When there is no dominating server, without loss of generality, we can assume that we have $\mu_{11} \geq \mu_{21}$ and $\mu_{22} \geq \mu_{12}$ (server 1 is faster at the first station and server 2 is faster at the second station), and when there is a dominating server, we can assume $\mu_{11} \geq \mu_{21}$ and $\mu_{12} \geq \mu_{22}$ (server 1 is faster at both stations). We characterize the optimal server assignment completely. Furthermore, we show that the optimal policy is unique subject to the interpretation that assigning a server to a station that is

blocked or starved is equivalent to idling the server.

3.4.1 Systems without a Dominating Server

We start our analysis with systems with specialized servers. The following proposition gives the optimal server assignment policy when there is no dominating server.

Proposition 3. *In a Markovian non-collaborative system with two stations and two servers, if we have* $\mu_{11} \geq \mu_{21}$ *and* $\mu_{22} \geq \mu_{12}$*, then the policy that assigns server* 1 *to the first station and server* 2 *to the second station at all times is optimal. Moreover, if* $\mu_{11} > \mu_{21}, \mu_{22} > \mu_{12}$ *then the optimal policy is unique in the class of Markovian stationary deterministic policies.*

Proof. The optimality of the dedicated policy that assigns server 1 to the first station and server 2 to the second station follows from Theorem 1. Also, under this optimal policy, all states are recurrent given that $\mu_{11} > \mu_{21}, \mu_{22} > \mu_{12}$, since we then must have $\mu_{11} > 0, \mu_{22} > 0$, and the optimal throughput is positive. To prove uniqueness, let us first eliminate the idling actions. If $s = 0$, then only the second station is starved, and assigning a server to the second station is the same as idling that server. Therefore actions a_{10} and a_{01} are equivalent to actions a_{12} and a_{21}, respectively. Moreover, actions that idle the server at the first station (i.e., a_{20}, a_{02}, a_{00}) result in zero throughput. Similarly if $s = B_1 + 2$, then only the first station is blocked and assigning a server to the first station is the same as idling that server. Therefore actions a_{02} and a_{20} are equivalent to actions a_{12} and a_{21}, respectively. Moreover, actions that idle the server at the second station (i.e., a_{10}, a_{01}, a_{00}) result in zero throughput. Thus, we do not have to consider idling actions in these states. Note that if a policy uses one of the actions $\{a_{00}, a_{10}, a_{01}, a_{20}, a_{02}\}$ in some state $s \in \{1, \ldots, B_1 + 1\}$, then the states $s - 1$ and $s + 1$ do not communicate, and the recurrent classes correspond to systems with a smaller buffer space. Let us denote the optimal policy given in Proposition 3 by π^*. Under policy π^*, the resulting Markov chain is a birth-death

process with birth rate of μ_{11}, death rate of μ_{22}, and the stationary distribution ρ, where $\rho(s) = \frac{\mu_{11}^s \mu_{22}^{B_1+2-s}}{\sum_{i=0}^{B_1+2} \mu_{11}^i \mu_{22}^{B_1+2-i}}$ for $s \in \{0, \ldots, B_1+2\}$. Thus, one can compute the long-run average throughput under the policy π^* as

$$T^{\pi^*}(B_1) = \frac{\sum_{i=0}^{B_1+1} \mu_{11}^{i+1} \mu_{22}^{B_1+2-i}}{\sum_{i=0}^{B_1+2} \mu_{11}^i \mu_{22}^{B_1+2-i}},$$

and show that it is strictly increasing in the buffer size B_1, since

$$T^{\pi^*}(B_1) - T^{\pi^*}(B_1-1) = \frac{\mu_{11}^{B_1+2} \mu_{22}^{B_1+2}}{\Big(\sum_{i=0}^{B_1+2} \mu_{11}^i \mu_{22}^{B_1+2-i}\Big)\Big(\sum_{i=0}^{B_1+1} \mu_{11}^i \mu_{22}^{B_1+1-i}\Big)} > 0.$$

Therefore, any policy that results in smaller recurrent classes cannot be optimal, the idling actions are strictly suboptimal, and we can reduce our action space to $\{a_{12}, a_{21}\}$.

Given that all states must be recurrent under an optimal policy, it is easy to see that the action a_{12} is unique optimal in the end states when $\mu_{11} > \mu_{21}$ and $\mu_{22} > \mu_{12}$ because the transition probabilities of the embedded discrete time Markov Chain out of states 0 and B_1+2 do not depend on the action we choose, and the sojourn times in these states are strictly smaller under action a_{12}.

Next, let us assume we have a policy π that uses action a_{21} at some state $s \in \{1, \ldots, B_1+1\}$. Note that if $\mu_{12} = 0$ or $\mu_{21} = 0$, policy π results in a recurrent class that corresponds to a system with smaller buffer size, and hence policy π is strictly suboptimal given that $\mu_{11} > \mu_{21}, \mu_{22} > \mu_{12}$. Hence we can assume both μ_{12}, μ_{21} are strictly positive for the rest of the proof. We will construct a randomized policy π' that is exactly the same as policy π except in state s, that has the same embedded chain as policy π, and has a strictly smaller expected sojourn time at state s.

First assume we have $\mu_{11}\mu_{12} \leq \mu_{21}\mu_{22}$, and policy π' uses action a_{12} with probability p and action a_{10} with probability $1-p$ at state s, where

$$p = \frac{\mu_{12}(\mu_{11}+\mu_{22})}{\mu_{22}(\mu_{12}+\mu_{21})}.$$

Note that $p \in (0, 1]$ due to the assumptions that $\mu_{11}\mu_{12} \leq \mu_{21}\mu_{22}$ and $\mu_{12} > 0, \mu_{21} > 0$. Then, under policy π', the transition probabilities out of state s will be the same as

under policy π ($P_{s,s-1} = \frac{\mu_{12}}{\mu_{12}+\mu_{21}}$, $P_{s,s+1} = \frac{\mu_{21}}{\mu_{12}+\mu_{21}}$) and the expected sojourn time at state s will become

$$\eta' = \frac{\mu_{21}}{\mu_{11}(\mu_{12}+\mu_{21})},$$

whereas the expected sojourn time under policy π is $\eta = \frac{1}{\mu_{12}+\mu_{21}}$, and $\eta' < \eta$ for $\mu_{11} > \mu_{21}$. Hence the long-run average throughput is strictly larger under policy π'.

When $\mu_{11}\mu_{12} > \mu_{21}\mu_{22}$, a similar randomized policy can be constructed using actions a_{12}, a_{02}, and $p = \frac{\mu_{21}(\mu_{11}+\mu_{22})}{\mu_{11}(\mu_{12}+\mu_{21})}$ at state s. Again the resulting transition probabilities will be unchanged and the mean sojourn time for state s will be strictly smaller due to our assumption $\mu_{22} > \mu_{12}$. Thus, there exist a randomized policy π' that has larger long-run average throughput than policy π. Theorem 9.1.8 of Puterman [56] implies that there must also exist a deterministic server assignment policy that performs at least as well as the randomized policy π'. Hence we conclude that if $\mu_{11} \geq \mu_{21}, \mu_{22} \geq \mu_{12}$ ($\mu_{11} > \mu_{21}, \mu_{22} > \mu_{12}$), then the action a_{21} is suboptimal (strictly suboptimal) at any state s. Therefore the policy given in Proposition 3 is the unique optimal policy when $\mu_{11} > \mu_{21}, \mu_{22} > \mu_{12}$. □

Remark 2. *For a two-station, two-server system with no dominating server, the optimality of the dedicated server assignment policy which assigns server* 1 *to the first station and server* 2 *to the second station also follows from the arguments used in the proof of uniqueness.*

3.4.2 Systems with a Dominating Server

When there is a dominating server, a more detailed approach is needed. First we let $(\delta)^\infty$ denote the policy corresponding to decision rule δ. Here a decision rule is a $(B_1 + 3)$-dimensional vector with components $\delta(s) \in A_s$ showing which action is chosen in state $s \in S$.

Let us define δ^i for $i \in \{0, 1, \ldots, B_1 + 3\}$ such that

$$\delta^i(s) = \begin{cases} a_{12} & \text{for } 0 \le s \le i-1, \\ a_{21} & \text{for } i \le s \le B_1+2. \end{cases}$$

Observe that the policy $(\delta^i)^\infty$ is a non-idling threshold policy that assigns server 1 to the first station and server 2 to the second station in states $\{0, \ldots, i-1\}$ (i.e., when the number of jobs that are processed at the first station but not at the second is less than i), and server 1 to the second station and server 2 to the first station in states $\{i, \ldots, B_1+2\}$ (i.e., when the number of jobs that are processed at the first station but not at the second is at least i). Thus, the faster server (server 1) divides his time between the two stations under the policy $(\delta^i)^\infty$ when $1 \le i \le B_1+2$. Also, the long-run average throughput function $T^{(\delta^i)^\infty}(B_1)$, $i \in \{0, 1, \ldots, B_1+3\}$ corresponding to the policy $(\delta^i)^\infty$ can be computed as:

$$T^{(\delta^i)^\infty}(B_1) = \frac{\mu_{12}^{B_1+3-i} \sum_{j=1}^{i-1} \mu_{11}^j \mu_{22}^{i-j} + \mu_{11}^i \sum_{j=0}^{B_1+2-i} \mu_{12}^{B_1+3-i-j} \mu_{21}^j}{\mu_{12}^{B_1+3-i} \sum_{j=1}^{i} \mu_{11}^{j-1} \mu_{22}^{i-j} + \mu_{11}^i \sum_{j=0}^{B_1+2-i} \mu_{12}^{B_1+2-i-j} \mu_{21}^j} \tag{2}$$

for $i \in \{1, \ldots, B_1+3\}$, and

$$T^{(\delta^0)^\infty}(B_1) = \frac{\sum_{j=0}^{B_1+1} \mu_{12}^{B_1+2-j} \mu_{21}^{j+1}}{\sum_{j=0}^{B_1+2} \mu_{12}^{B_1+2-j} \mu_{21}^j}. \tag{3}$$

Note that we use the convention that summation over an empty set is zero and $0^0 = 1$. Consider the function:

$$f(i) = (\mu_{11} - \mu_{21}) \mu_{22}^{i-1} \sum_{j=0}^{B_1+2-i} \mu_{12}^{B_1+3-i-j} \mu_{21}^j + (\mu_{22} - \mu_{12}) \mu_{21}^{B_1+3-i} \sum_{j=0}^{i-2} \mu_{11}^{j+1} \mu_{22}^{i-j-2}$$

for $i \in \{1, \ldots, B_1+3\}$. It can be shown that $T^{(\delta^i)^\infty}(B_1) - T^{(\delta^{i-1})^\infty}(B_1)$ is a positive multiple of the simpler function $f(i)$. While this provides a useful interpretation of our results, it is not needed to prove them.

We now present a lemma and a corollary that describe some useful properties of the function f.

Lemma 1. *The function $f(i)$ is nonnegative for $i = 1$ and nonpositive for $i = B_1+3$. Also it is nonincreasing in i, i.e., $f(i+1) - f(i) \le 0$ for $i \in \{1, \ldots, B_1+2\}$.*

Proof. First we prove that $f(1)$ is nonnegative and $f(B_1+3)$ is nonpositive:

$$f(1) = (\mu_{11} - \mu_{21}) \sum_{j=0}^{B_1+1} \mu_{12}^{B_1+2-j} \mu_{21}^{j} \geq 0,$$

and

$$f(B_1+3) = (\mu_{22} - \mu_{12}) \sum_{j=0}^{B_1+1} \mu_{11}^{j+1} \mu_{22}^{B_1+1-j} \leq 0,$$

since $\mu_{11} \geq \mu_{21}$ and $\mu_{22} \leq \mu_{12}$.

Now, consider the difference $f(i+1) - f(i)$ for $i \in \{1, \ldots, B_1+2\}$:

$$\begin{aligned} f(i+1) - f(i) = {} & \mu_{22}^{i-1}(\mu_{11} - \mu_{21})(\mu_{22} - \mu_{12}) \sum_{j=0}^{B_1+1-i} \mu_{12}^{B_1+2-i-j} \mu_{21}^{j} \\ & + \mu_{21}^{B+2-i} \mu_{11}(\mu_{11} - \mu_{21})(\mu_{22} - \mu_{12}) \sum_{j=0}^{i-2} \mu_{11}^{i-2-j} \mu_{22}^{j} \\ & + \mu_{21}^{B_1+2-i} \mu_{11}(\mu_{22} - \mu_{12}) \mu_{22}^{i-1} + \mu_{22}^{i-1}(\mu_{21} - \mu_{11}) \mu_{12} \mu_{21}^{B_1+2-i} \\ \leq {} & 0, \end{aligned}$$

since $\mu_{11} \geq \mu_{21}$ and $\mu_{12} \geq \mu_{22}$ imply that $(\mu_{22} - \mu_{12})(\mu_{11} - \mu_{21}) \leq 0$. □

Corollary 1. *The set*

$$S^* = \{s \in S \backslash \{0\} : \ f(s) \geq 0, \ f(s+1) \leq 0\}$$

is non-empty. Moreover, if there are multiple elements in S^*, then they are consecutive states.

Proof. By Lemma 1, we have $f(1) \geq 0$ and $f(B_1+3) \leq 0$. Therefore there exists s^* at which $f(s^*) \geq 0$ and $f(s^*+1) \leq 0$ and S^* is nonempty. Moreover, the function $f(i)$ is nonincreasing in i, i.e., $f(i+1) - f(i) \leq 0$ for $i \in \{1, \ldots, B_1+3\}$, hence the elements of S^* are consecutive states. □

In the following theorem, we use the function f to state our result that characterizes the optimal policy. Furthermore, we show that the optimal policy is unique

subject to the interpretation that assigning a server to a station that is blocked or starved is equivalent to idling the server.

Theorem 2. *If server* 1 *is faster than server* 2 *at both stations, then the policy* $(\delta^{s^*})^\infty$ *is optimal, where* $s^* \in S^*$. *Furthermore, it is the unique optimal policy in the class of Markovian stationary deterministic policies if* $f(s^*) > 0, f(s^*+1) < 0$.

Proof. We know that an optimal Markovian stationary deterministic policy exists by Theorem 9.1.8 of Puterman [56] since the state and action spaces are finite. We use Policy Iteration to show that the policy defined in the theorem is optimal. Let us choose the initial decision rule $\delta_0 = \delta^{s^*}$ as in Theorem 2, and let π_0 denote the corresponding policy. We can assume that $\mu_{11} > 0$ and $\mu_{12} > 0$ because otherwise there is a station at which neither server can work and the throughput of all policies is zero. Note that if $\mu_{11} = 0$ or $\mu_{12} = 0$, then $f(i) = 0$ for all $i \in \{1, \ldots, B_1 + 3\}$, and thus the uniqueness condition given in Theorem 2 does not hold. Also, we assume at least one of μ_{21}, μ_{22} is nonzero, because if $\mu_{21} = \mu_{22} = 0$, there is only one server to work at two stations and any Markovian deterministic policy with non-zero long-run average throughput, including the one given in Theorem 2, is optimal. In this case, $f(1) = \mu_{11}\mu_{12}^{B_1+2} > 0$, $f(B_1+3) = -\mu_{11}^{B_1+2}\mu_{12} < 0$, and $f(i) = 0$ for $1 < i < B_1 + 3$, and the uniqueness condition given in Theorem 2 does not hold.

We start the Policy Iteration algorithm for a communicating model. Let r_{δ_0} and P_{δ_0} denote the corresponding reward vector and probability transition matrix for the decision rule δ_0, respectively. Without loss of generality, the uniformization constant can be taken as 1. We have

$$P_{\delta_0}(s,s') = \begin{cases} 1-\mu_{11} & \text{for } \ s=0, s'=0, \\ \mu_{11} & \text{for } \ s=0, s'=1, \\ 0 & \text{for } \ s=0, s'\geq 2, \\ \mu_{22} & \text{for } 1\leq s\leq s^*-1, s'=s-1, \\ 1-\mu_{22}-\mu_{11} & \text{for } 1\leq s\leq s^*-1, s'=s, \\ \mu_{11} & \text{for } 1\leq s\leq s^*-1, s'=s+1, \\ 0 & \text{for } 1\leq s\leq s^*-1, s'>s+1 \text{ or } s'<s-1, \\ \mu_{12} & \text{for } s^*\leq s\leq B_1+1, s'=s-1, \\ 1-\mu_{12}-\mu_{21} & \text{for } s^*\leq s\leq B_1+1, s'=s, \\ \mu_{21} & \text{for } s^*\leq s\leq B_1+1, s'=s+1, \\ 0 & \text{for } s^*\leq s\leq B_1+1, s'>s+1 \text{ or } s'<s-1, \\ \mu_{12} & \text{for } s=B_1+2, s'=B_1+1, \\ 1-\mu_{12} & \text{for } s=B_1+2, s'=B_1+2, \\ 0 & \text{for } s=B_1+2, s'\leq B_1; \end{cases}$$

$$r_{\delta_0}(s) = \begin{cases} 0 & \text{for } s=0, \\ \mu_{22} & \text{for } 1\leq s\leq s^*-1, \\ \mu_{12} & \text{for } s^*\leq s\leq B_1+2. \end{cases}$$

Note that $\mu_{11}>0$ and $\mu_{12}>0$ implies that the decision rule δ_0 yields a unichain structure. We can solve the following equation to find g_0 and h_0:

$$r_{\delta_0} - g_0 e + (P_{\delta_0} - I)h_0 = 0, \tag{4}$$

where e is the unit vector and $h_0(0)=0$. In particular, $g_0 = T^{(\delta^{s^*})^\infty}(B_1)$, where the function T is defined in equations (2) and (3).

For $s\leq s^*$:

$$h_0(s) = g_0 \sum_{i=0}^{s-1}(s-i)\frac{\mu_{22}^i}{\mu_{11}^{i+1}} - \sum_{i=0}^{s-2}(s-1-i)\frac{\mu_{22}^{i+1}}{\mu_{11}^{i+1}} \ .$$

For $s > s^*$:

$$h_0(s) = g_0\Bigg(\sum_{i=0}^{s^*-1}(s^*-i)\frac{\mu_{22}^i}{\mu_{11}^{i+1}}\frac{\mu_{12}^{s-s^*}}{\mu_{21}^{s-s^*}} + \frac{(\mu_{22}-\mu_{12})}{\mu_{21}}\sum_{i=0}^{s-s^*-1}\frac{\mu_{12}^i}{\mu_{21}^i}\sum_{j=0}^{s^*-2}(s^*-1-j)\frac{\mu_{22}^j}{\mu_{11}^{j+1}}$$
$$+ \sum_{i=0}^{s-s^*-1}(s-i)\frac{\mu_{12}^i}{\mu_{21}^{i+1}} + \frac{(\mu_{21}-\mu_{11})}{\mu_{21}}\sum_{i=0}^{s-s^*-1}\frac{\mu_{12}^i}{\mu_{21}^i}\sum_{j=1}^{s^*-1}(s^*-j)\frac{\mu_{22}^j}{\mu_{11}^{j+1}}\Bigg)$$
$$- \sum_{i=0}^{s-s^*-1}(s-s^*-i)\frac{\mu_{12}^{i+1}}{\mu_{21}^{i+1}} + \frac{(\mu_{22}-\mu_{12})}{\mu_{21}}\sum_{i=0}^{s-s^*-1}\frac{\mu_{12}^i}{\mu_{21}^i}\sum_{j=0}^{s^*-3}(s^*-2-j)\frac{\mu_{22}^{j+1}}{\mu_{11}^{j+1}}$$
$$+ \sum_{i=0}^{s^*-2}(s^*-1-i)\frac{\mu_{22}^{i+1}}{\mu_{11}^{i+1}}\frac{\mu_{12}^{s-s^*}}{\mu_{21}^{s-s^*}} + (s^*-1)\frac{\mu_{22}}{\mu_{21}}\sum_{i=0}^{s-s^*-1}\frac{\mu_{12}^i}{\mu_{21}^i}$$
$$+ \frac{(\mu_{21}-\mu_{11})}{\mu_{21}}\sum_{i=0}^{s-s^*-1}\frac{\mu_{12}^i}{\mu_{21}^i}\sum_{j=0}^{s^*-2}(s^*-1-j)\frac{\mu_{22}^{j+1}}{\mu_{11}^{j+1}}.$$

The Policy Iteration algorithm terminates, proving that π_0 is optimal, if the following is true for all states $s \in \{0, 1, \ldots, B_1+2\}$ and for all actions $a \in A_s$ other than $\delta_0(s)$:

$$\Delta(s,a) = r(s,a) + \sum_{j\in S} p(j|s,a)h_0(j) - r(s,\delta_0(s)) - \sum_{j\in S} p(j|s,\delta_0(s))h_0(j) \leq 0. \quad (5)$$

We will examine states $s \in S$ separately for $s < s^*$ and $s \geq s^*$ as our decision rule δ_0 takes different actions for states $s < s^*$ and $s \geq s^*$. We first show that the inequality (5) holds for non-idling actions (i.e., a_{21} when $0 \leq s < s^*$, and a_{12} when $s^* \leq s \leq B_1+2$).

Define

$$\Gamma = \mu_{12}^{B_1+3}\sum_{i=0}^{s^*-1}\mu_{11}^i\mu_{22}^{s^*-1-i} + \mu_{11}^{s^*}\sum_{i=0}^{B_1+2-s^*}\mu_{12}^{s^*+i}\mu_{21}^{B_1+2-s^*-i}.$$

Note that $\Gamma > 0$ under our assumptions that $\mu_{11} > 0, \mu_{12} > 0$. If $0 \leq s < s^*$, the

right-hand side of (5) with $a = a_{21}$ becomes $\Delta(s, a_{21}) = \frac{\Gamma_1(s)}{\Gamma}$, where

$$\begin{aligned}\Gamma_1(s) = (\mu_{21} - \mu_{11})\mu_{11}^{s^*-s-1}\mu_{22}^{s} \sum_{i=0}^{B_1+1-s^*} \mu_{12}^{s^*+1+i}\mu_{21}^{B_1+2-s^*-i} \\ + (\mu_{12} - \mu_{22})\mu_{12}^{s^*}\mu_{21}^{B_1+3-s^*} \sum_{i=0}^{s-1} \mu_{11}^{s^*-s+i}\mu_{22}^{s-1-i} \\ + (\mu_{21} - \mu_{11})\mu_{12}^{B_1+3} \sum_{i=0}^{s^*-s-1} \mu_{11}^{i}\mu_{22}^{s^*-1-i}.\end{aligned}$$

It follows with some algebra that $\Gamma_1(s^* - 1)$ is a negative multiple of $f(s^*)$, in particular, $\Gamma_1(s^* - 1) = -f(s^*)\mu_{12}^{s^*}$. Since $f(s^*) \geq 0$, $\Gamma_1(s^* - 1)$ is nonpositive and $\Delta(s^* - 1, a_{21}) \leq 0$, proving our claim at state $s^* - 1$. Next we prove that $\Gamma_1(s)$ is nondecreasing in s by showing $\Gamma_1(s-1) - \Gamma_1(s) \leq 0$ for $1 < s < s^*$. We have:

$$\Gamma_1(s-1) - \Gamma_1(s) = -\mu_{11}^{s^*-s-1}\mu_{22}^{s-1}(\mu_{11}\mu_{12} - \mu_{21}\mu_{22})\Big((\mu_{11} - \mu_{21})\mu_{12}^{s^*} \sum_{i=0}^{B_1+1-s^*} \mu_{12}^{B_1+2-s^*-i}\mu_{21}^{i} + \mu_{11}\mu_{12}^{B_1+2}\Big).$$

It follows from our assumptions $\mu_{11} \geq \mu_{21}$ and $\mu_{12} \geq \mu_{22}$ that the above expression is nonpositive. Hence $\Gamma_1(s)$ is nondecreasing in s, and we have $\Delta(s, a_{21}) \leq 0$ for all $0 \leq s < s^*$. Furthermore, the inequality (5) is strict for $0 \leq s < s^*$ unless $f(s^*) = 0$ (because $\Gamma_1(s^* - 1) < 0$).

On the other hand, if $s^* \leq s \leq B_1 + 2$, the right-hand side of expression (5) with $a = a_{12}$ becomes $\Delta(s, a_{12}) = \frac{\Gamma_2(s)}{\Gamma}$, where

$$\begin{aligned}\Gamma_2(s) = \mu_{11}^{s^*}(\mu_{22} - \mu_{12}) \sum_{i=0}^{s-s^*-1} \mu_{12}^{s^*+i}\mu_{21}^{B_1-s^*-i+2} \\ + (\mu_{11} - \mu_{21})\mu_{22}^{s^*} \sum_{i=0}^{B_1+1-s} \mu_{21}^{i}\mu_{12}^{B_1+2-i} \\ + (\mu_{22} - \mu_{12})\mu_{12}^{s}\mu_{21}^{B_1+2-s} \sum_{i=0}^{s^*-1} \mu_{11}^{i+1}\mu_{22}^{s^*-1-i}.\end{aligned}$$

Evaluating $\Gamma_2(s)$ at $s = s^*$, one can show that $\Gamma_2(s^*)$ is a positive multiple of $f(s^*+1)$, namely $\Gamma_2(s^*) = f(s^* + 1)\mu_{12}^{s^*}$. By definition of s^*, we know that $f(s^* + 1) \leq 0$.

Therefore we have $\Gamma_2(s^*) \leq 0$ and $\Delta(s^*, a_{12}) \leq 0$ proving our claim for $s = s^*$. We prove that $\Gamma_2(s)$ is non-increasing in s by looking at $\Gamma_2(s) - \Gamma_2(s+1)$ for $s^* \leq s < B_1 + 1$ and showing it to be non-negative. We have:

$$\Gamma_2(s) - \Gamma_2(s+1) = -\mu_{12}^{s}\mu_{21}^{B_1+1-s}(\mu_{21}\mu_{22} - \mu_{11}\mu_{12})\Big((\mu_{12} - \mu_{22})\sum_{i=0}^{s^*-2}\mu_{22}^{i}\mu_{11}^{s^*-1-i} + \mu_{12}\mu_{22}^{s^*-1}\Big).$$

Due to our assumptions $\mu_{11} \geq \mu_{21}$ and $\mu_{12} \geq \mu_{22}$, it follows that $\Gamma_2(s) - \Gamma_2(s+1) \geq 0$. Therefore, $\Delta(s^*, a_{12}) \leq 0$ implies that $\Delta(s, a_{12}) \leq 0$ for all states s such that $s^* \leq s \leq B_1 + 2$. Also, inequality (5) is strict for all states $s^* \leq s \leq B_1 + 2$ unless $f(s^* + 1) = 0$ (because $\Gamma_2(s^*) < 0$).

Since inequality (5) holds for all states $s \in \{0, \ldots, B_1 + 2\}$ and non-idling actions, we have shown the policy $(\delta_0)^\infty$ is optimal among non-idling policies.

From the arguments given in the proof of Proposition 3, it follows that we do not need to consider idling actions for $s = 0$ and $s = B_1 + 2$ (because idling actions are either equivalent to non-idling actions or they are strictly suboptimal). Here we use induction on B_1 to show that a policy that uses an idling action in any of the states $s \in \{1, \ldots, B_1 + 1\}$ cannot be optimal.

If $B_1 = 0$, the state space becomes $S = \{0, 1, 2\}$ and it is enough to prove that idling actions are strictly suboptimal in state $s = 1$. Note that the transition probabilities of the embedded discrete time Markov Chain out of states $s = 0$ and $s = 2$ does not depend on the action we choose, and the sojourn times in these states are smaller under actions a_{12}, a_{21}, respectively. Thus, these actions are optimal in $s = 0$ and $s = 2$. Let δ_a denote the decision rule that uses an idling action $a \in \{a_{00}, a_{10}, a_{01}, a_{20}, a_{02}\}$ in $s = 1$, and non-idling actions a_{12}, a_{21} in $s = 0$, $s = 2$, respectively. It is easy to see that decision rule $\delta_{a_{00}}$ is strictly suboptimal since it results in zero throughput. Note that $\delta_0 \in \{\delta^1, \delta^2\}$ and define $\kappa_{a,i} = T^{(\delta^i)^\infty}(0) - T^{(\delta_a)^\infty}(0)$ for $i \in S \backslash \{0\} = \{1, 2\}$. We

have

$$\kappa_{a_{10},1} = \frac{\mu_{11}\mu_{12}^2\mu_{21}}{(\mu_{11}+\mu_{12})(\mu_{12}^2+\mu_{11}\mu_{12}+\mu_{11}\mu_{21})},$$
$$\kappa_{a_{02},1} = \frac{(\mu_{12}-\mu_{22})\mu_{11}^2\mu_{12}+(\mu_{12}-\mu_{22})\mu_{11}^2\mu_{21}+\mu_{11}\mu_{12}\mu_{21}\mu_{22}}{(\mu_{11}+\mu_{22})(\mu_{12}^2+\mu_{11}\mu_{12}+\mu_{11}\mu_{21})},$$
$$\kappa_{a_{01},1} = \frac{(\mu_{11}-\mu_{21})\mu_{12}^3+\mu_{11}\mu_{12}^2\mu_{21}}{(\mu_{12}+\mu_{21})(\mu_{12}^2+\mu_{11}\mu_{12}+\mu_{11}\mu_{21})},$$
$$\kappa_{a_{20},1} = \frac{\mu_{11}\mu_{12}^2\mu_{21}}{(\mu_{12}+\mu_{12})(\mu_{12}^2+\mu_{11}\mu_{12}+\mu_{11}\mu_{21})},$$
$$\kappa_{a_{10},2} = \frac{\mu_{11}^2\mu_{12}\mu_{22}}{(\mu_{11}+\mu_{12})(\mu_{11}^2+\mu_{11}\mu_{12}+\mu_{12}\mu_{22})},$$
$$\kappa_{a_{02},2} = \frac{\mu_{11}^3(\mu_{12}-\mu_{22})+\mu_{11}^2\mu_{12}\mu_{22}}{(\mu_{11}+\mu_{22})(\mu_{11}^2+\mu_{11}\mu_{12}+\mu_{12}\mu_{22})},$$
$$\kappa_{a_{01},2} = \frac{(\mu_{11}-\mu_{21})\mu_{12}^2\mu_{22}+(\mu_{11}-\mu_{21})\mu_{11}\mu_{12}^2+\mu_{11}\mu_{12}\mu_{21}\mu_{22}}{(\mu_{12}+\mu_{21})(\mu_{11}^2+\mu_{11}\mu_{12}+\mu_{12}\mu_{22})},$$
$$\kappa_{a_{20},2} = \frac{\mu_{11}^2\mu_{12}\mu_{22}}{(\mu_{11}+\mu_{12})(\mu_{11}^2+\mu_{11}\mu_{12}+\mu_{12}\mu_{22})}.$$

We consider three cases. First, if $\mu_{21} > 0, \mu_{22} > 0$, we have $\kappa_{a,i} > 0$ for all $i \in \{1,2\}$ and $a \in \{a_{10}, a_{01}, a_{20}, a_{02}\}$. Similarly, if $\mu_{21} > 0, \mu_{22} = 0$, then $\kappa_{a,1} > 0$ for all $a \in \{a_{10}, a_{01}, a_{20}, a_{02}\}$. Finally, if $\mu_{21} = 0, \mu_{22} > 0$, we have $\kappa_{a,2} > 0$ for all $a \in \{a_{10}, a_{01}, a_{20}, a_{02}\}$. Therefore, for any idling policy $(\delta_a)^\infty$, there exists a non-idling policy $(\delta^i)^\infty$ that performs strictly better, and thus idling actions are strictly suboptimal for $B_1 = 0$ in all three cases.

Assume now that the non-idling decision rule δ_0 is optimal among all possible decision rules for all buffer sizes $B_1 \leq B_1'$. (Note that δ_0 depends on the buffer size B_1, but we suppress this in our notation.) For buffer size $B_1' + 1$, assume there exists an optimal decision rule δ' that uses an idling action at some state $s \in \{1, \ldots, B_1' + 2\}$. Under decision rule δ', states $s-1$ and $s+1$ do not communicate and the resulting recurrent classes correspond to systems with buffer size strictly smaller than $B_1' + 1$. Let $B'' < B_1' + 1$ denote the buffer size for any one of the resulting systems. By our assumption, δ_0 is optimal for this system, hence the (constant) long-run average throughput achieved by δ' must be equal to that of δ_0. We now show that this leads

to a contradiction.

We have $T^{\delta^i}(B_1+1) - T^{\delta^i}(B_1) = \frac{\nu_1(i,B_1)}{\nu_2(i,B_1)\nu_2(i,B_1+1)}$ for all $i \in \{1,\ldots,B_1+2\}$, where

$$\nu_1(i,k) = \mu_{11}^i \mu_{12}^{k+3+i} \mu_{21}^{k+3-i} \Big((\mu_{12}-\mu_{22}) \sum_{j=0}^{i-2} \mu_{22}^j \mu_{11}^{i-1-j} + \mu_{12}\mu_{22}^{i-1} \Big),$$

$$\nu_2(i,k) = \mu_{11}^i \sum_{j=0}^{k+2-i} \mu_{12}^{i+j} \mu_{21}^{k+2-i-j} + \mu_{12}^{k+3} \sum_{j=0}^{i-1} \mu_{11}^j \mu_{22}^{i-1-j}.$$

If $\mu_{21} > 0, \mu_{22} > 0$, $T^{\delta^i}(B_1+1) - T^{\delta^i}(B_1) > 0$ for all $i \in \{1,\ldots,B_1+2\}$, implying that the throughput achieved by the decision rule δ_0 is strictly increasing in the buffer size. Similarly, if $\mu_{21} > 0, \mu_{22} = 0$, we have $f(1) \geq 0$, $f(i) < 0$ for $i \in \{2,\ldots,B_1+3\}$ and $s^* = 1$, $\delta_0 = \delta^1$ for all buffer sizes, and the throughput achieved by the decision rule δ_0 is strictly increasing in buffer size since $T^{(\delta^1)^\infty}(B_1+1) - T^{(\delta^1)^\infty}(B_1) > 0$. Finally, if $\mu_{21} = 0, \mu_{22} > 0$, we have $f(i) > 0$ for $i \in \{1,\ldots,B_1+2\}$, $f(B_1+3) \leq 0$, and $s^* = B_1+2$. Thus, $\delta_0 = \delta^{B_1+2}$ for buffer size B_1 and $\delta_0 = \delta^{B_1+3}$ for buffer size B_1+1. We have $T^{(\delta^{B_1+3})^\infty}(B_1+1) - T^{(\delta^{B_1+2})^\infty}(B_1) = \frac{\nu_3(B_1)}{\nu_4(B_1)\nu_4(B_1+1)} > 0$, where

$$\nu_3(k) = \mu_{11}^{k+3} \mu_{12} \mu_{22}^{k+2},$$

$$\nu_4(k) = \mu_{11}^{k+2} + \mu_{12} \sum_{i=0}^{k+1} \mu_{11}^i \mu_{22}^{k+1-i}.$$

In all three cases, the throughput achieved by the decision rule δ_0 is strictly increasing in the buffer size, and we must have $T^{(\delta')^\infty}(B_1'+1) = T^{(\delta_0)^\infty}(B'') < T^{(\delta_0)^\infty}(B_1'+1)$, contradicting the assumption that the decision rule δ' is optimal. We conclude that no policy that uses an idling action in a state $s \in \{1,\ldots,B_1'+2\}$ can be optimal for buffer size $B_1'+1$. By induction, it follows that idling actions are strictly suboptimal and can be eliminated for all buffer sizes. Therefore, the decision rule δ_0 is optimal in the class of Markovian stationary deterministic policies.

We have shown that policies with idling actions are strictly suboptimal. To prove uniqueness among non-idling policies, we use a similar approach to Ayhan and Andradóttir [9] and consider a non-idling decision rule δ' that differs from δ_0 in at least

one state $s \in S$. Let us define,

$$u = P_{\delta'} g_0 e - g_0 e = 0,$$

$$v = r_{\delta'} + (P_{\delta'} - I)h_0 - g_0 e = r_{\delta'} + P_{\delta'} h_0 - (r_{\delta_0} + P_{\delta_0} h_0),$$

where we have used equation (4). It is shown in the proof of optimality above that $v(s) \leq 0$ for all $s \in S$ and that if $f(s^*) > 0, f(s^*+1) < 0$, we must have

$$v(s) < 0 \text{ for all } s \in S \text{ with } \delta'(s) \neq \delta_0(s). \tag{6}$$

It is easy to see that if both μ_{21}, μ_{22} are positive, then P_{δ_0} must be irreducible since δ_0 is non-idling. Moreover, $\delta_0 = \delta^{B_1+2}$ and $\delta_0 = \delta^1$ also result in irreducible transition matrices when $\mu_{21} = 0, \mu_{22} > 0$ and $\mu_{21} > 0, \mu_{22} = 0$, respectively. Hence, given that at least one of μ_{21}, μ_{22} is nonzero, P_{δ_0} is irreducible.

Since P_{δ_0} is irreducible and δ' differs from δ_0 in at least one state, then it must differ from δ_0 in at least one state $s_0 \in S$ that is recurrent under δ'. Let g' denote the (possibly state dependent) throughput of the stationary policy $(\delta')^\infty$ and define $\Delta g = g' - g_0 e$. Also let $P^*_{\delta'}$ denote the limiting matrix of $P_{\delta'}$. Suppose $P_{\delta'}$ has n recurrent classes, and partition $P_{\delta'}$ such that P_i for $i \in \{1, 2, \ldots, n\}$ corresponds to transitions within recurrent class i. Also partition g', Δg, and $P^*_{\delta'}$ in a manner consistent with the partition of $P_{\delta'}$. Lemma 9.2.5 of Puterman [56] states that $\Delta g_i = P^*_i v_i$. Using this lemma and equation (6), we conclude $g'(s_0) - g_0 < 0$. Thus the decision rule δ' cannot be optimal, proving that $(\delta_0)^\infty$ is the unique optimal policy if $f(s^*) > 0, f(s^*+1) < 0$. □

Remark 3. *The uniqueness condition $f(s^*) > 0, f(s^*+1) < 0$ given in Theorem 2 implies that the set S^* has a single element s^*. However, the converse is not true since if $s^* = 1$, $f(1) = 0$, and $f(2) < 0$, or if $s^* = B_1 + 2$, $f(B_1 + 2) > 0$, and $f(B_1 + 3) = 0$, the uniqueness condition does not hold and $S^* = \{s^*\}$. Moreover,*

if $\mu_{11} = \mu_{21}$*, then there are multiple optimal actions in state* $s = 0$*. Similarly, if* $\mu_{12} = \mu_{22}$*, then there are multiple optimal actions in state* $s = B_1 + 2$*. Thus, the optimal policy is not unique unless* $\mu_{11} > \mu_{21}, \mu_{12} > \mu_{22}$*.*

3.4.3 Systems with Non-Exponential Service Requirements

When the service requirements are exponentially distributed, the threshold policy $(\delta^{s^*})^\infty$ defined in Theorem 2 is optimal for systems with two stations. We now evaluate the performance of the policy $(\delta^{s^*})^\infty$ for systems with two stations and service requirements that are not exponentially distributed. In particular, we consider systems where each service requirement follows a $Uniform(0, 2)$ distribution and the service requirement at station j is depleted at the service rate μ_{ij} when server i is assigned to station j.

For a non-Markovian system, we use the same state definition as in our formulation for systems with exponential service requirements. Thus, we only consider the server assignment policies that reassigns the servers after a service completion, and refer to these policies as *service dependent* (SD). For the set of service rates described in Table 1, we compare the long-run average throughput attained by the policy $(\delta^{s^*})^\infty$ to the long-run average throughput attained by the best SD policy. Note that when the service requirements are not exponentially distributed the best SD policy is not necessarily optimal. We also use the best dedicated policy and the best dedicated policy in a Markovian system with the same service rates μ_{ij} as benchmarks.

Table 1: Service rates for systems with uniformly distributed service requirements

Instance	μ_{11}	μ_{12}	μ_{21}	μ_{22}	Instance	μ_{11}	μ_{12}	μ_{21}	μ_{22}
1	3.00	4.00	1.00	2.00	2	2.00	4.00	1.00	3.00
3	4.00	3.00	2.00	1.00	4	4.00	2.00	3.00	1.00

For each set of service rates, the policies were simulated with buffer sizes $B_1 \in \{1, 5, 10\}$. Each instance were simulated for 20 replications of length 200,000 time units, with the first 10,000 time units of each replication being truncated. Table 2

Table 2: Throughput values for systems with $N = 2$ and Uniform[0,2] service requirements

Instance 1	Best SD	$(\delta^{s^*})^\infty$	Best Ded.-Uniform	Best Ded.-Exp
$B_1 = 1$	2.3464±0.0005	2.3422±0.0007	1.9216±0.0005	1.9216±0.0005
$B_1 = 5$	2.4957±0.0006	2.4900±0.0005	1.9997±0.0009	1.9997±0.0009
$B_1 = 10$	2.5027±0.0002	2.4987±0.0006	2.0003±0.0007	2.0003±0.0007
Instance 2	Best SD	$(\delta^{s^*})^\infty$	Best Ded.-Uniform	Best Ded.-Exp
$B_1 = 1$	1.9404±0.0006	1.9401±0.0008	1.9217±0.0007	1.9217±0.0007
$B_1 = 5$	2.0011±0.0004	2.0003±0.0007	1.9994±0.0010	1.9994±0.0010
$B_1 = 10$	2.0039±0.0003	2.0004±0.0008	1.9999±0.0008	1.9999±0.0008
Instance 3	Best SD	$(\delta^{s^*})^\infty$	Best Ded.-Uniform	Best Ded.-Exp
$B_1 = 1$	2.3465±0.0005	1.9980±0.0005	1.9215±0.0008	1.9215±0.0008
$B_1 = 5$	2.4960±0.0004	2.4874±0.0006	1.9985±0.0009	1.9985±0.0009
$B_1 = 10$	2.5032±0.0002	2.5005±0.0006	2.0000±0.0008	2.0000±0.0008
Instance 4	Best SD	$(\delta^{s^*})^\infty$	Best Ded.-Uniform	Best Ded.-Exp
$B_1 = 1$	1.9407±0.0005	1.5975±0.0007	1.9216±0.0004	1.9216±0.0004
$B_1 = 5$	2.0016±0.0006	1.9967±0.0007	1.9997±0.0006	1.99997±0.0006
$B_1 = 10$	2.0046±0.0003	1.9997±0.0008	2.0012±0.0009	2.0012±0.0009

shows the 95% confidence intervals for the average throughput of simulated policies.

The simulation results demonstrate that the structure of the best SD policy is insensitive to the service requirement distribution since a threshold policy performed best in all simulated instances. We observe that in general, the best SD policy has a lower switching threshold than s^* calculated for the Markovian system. Our results also suggest that the policy $(\delta^{s^*})^\infty$ has good performance even in systems with non-exponential service requirement distributions. For instances 1 and 2, the policy $(\delta^{s^*})^\infty$ achieves more than 99% of the throughput attained by the best SD policy for all buffer sizes $B_1 \in \{1, 5, 10\}$. For instances 3 and 4, the performance gap between the best SD policy and $(\delta^{s^*})^\infty$ is 15% and 17% when $B_1 = 1$. The relatively low performance of $(\delta^{s^*})^\infty$ in instances 3 and 4 is expected, since in these instances, rate of the slower server at the second station is significantly lower than the rate of the faster server, although the performance of the servers at the first station are comparable. This requires the serves to be switched at an even lower threshold, resulting in the low performance of threshold value s^*. However, the performance difference for different

threshold values decreases as the buffer size grows, and the gap is again less than 1% for the larger buffer sizes $B_1 = 5$ and $B_1 = 10$. Overall, the optimal policy in Markovian systems can be used as a near-optimal heuristic when the service times are non-exponential.

3.5 Collaborative vs. Non-Collaborative Systems

In this section, we compare collaborative and non-collaborative systems and provide insights on the benefits of collaboration in flexible systems. We also use the dedicated policies as benchmarks to evaluate the benefits of flexibility.

For non-collaborative systems with two stations and two servers, Proposition 3 shows that a dedicated server assignment policy is optimal when there is no dominating server and Theorem 2 shows that the optimal server assignment policy is non-idling and threshold type when there is a dominating server. For collaborative systems with two stations and two servers, Andradóttir et al. [11] characterized the optimal policy to be composed of a primary assignment and a contingency plan. In particular, if $\mu_{11}\mu_{22} - \mu_{21}\mu_{12} \geq 0$, the optimal policy assigns server 1 to station 1 and server 2 to station 2 unless station 1 is blocked or station 2 is starved (i.e., primary assignment), and assigns both servers to station 1 (station 2) when station 2 (station 1) is starved (blocked) (i.e., contingency plan). In a collaborative system, blocking and starvation of a station can be resolved relatively quickly since the service rate at a station can be increased by a greater amount by assigning multiple servers to a station. Thus, the threshold structure of the optimal policy in the non-collaborative setting disappears once collaboration is allowed.

For systems with more than two stations, the optimal policy is uncharacterized in general, regardless of the collaboration structure. We compare the optimal policies in collaborative and non-collaborative settings to evaluate the loss in the maximum attainable throughput due to lack of collaboration. Note that in both settings

the servers are flexible. We also evaluate the improvement in the long-run average throughput due to flexibility in contrast to dedicated server assignment policies.

In many manufacturing systems, tasks on which a worker has to work are similar in nature, and processing times are dependent on worker's skill level. For instance, in bucket brigade manufacturing, a worker has the same speed for each of the tasks he needs to perform, but workers vary in terms of how fast they can work [19]. In this section, we consider systems with homogeneous tasks where the service rates do not depend on the station but only on the server, i.e., the service rates $\mu_{ij} = \mu_i$ for $i \in \{1, \ldots, N\}$, $j \in \{1, \ldots, N\}$, as well as systems with non-homogeneous tasks and general service rates μ_{ij}.

Theorem 2 shows that if there is a dominant server, the optimal policy utilizes the dominant server at both stations. When the tasks are homogeneous, the two dedicated server assignment policies yield the same long-run average throughput, and without cross-training, the maximum attainable throughput in a two-station non-collaborative system is bounded above by the rate of the slower server:

$$T^{(\delta^0)^\infty} = T^{(\delta^{B_1+3})^\infty} = \mu_2 \left(\frac{\sum_{i=0}^{B_1+1} \mu_2^i \mu_1^{B_1+2-i}}{\sum_{i=0}^{B_1+2} \mu_2^i \mu_1^{B_1+2-i}} \right) \leq \mu_2.$$

On the other hand, if the servers are cross-trained, the maximum attainable throughput $T^{(\delta^{s^*})^\infty}$ is bounded below by $\mu_1/2$. Hence the improvement in the long-run average throughput due to cross-training increases without a bound as the service rate of the faster server increases, and cross-training can improve the maximum attainable throughput even if the servers are non-collaborating.

In a two-station collaborative system, the optimal policy moves the servers away from their primary stations to avoid idleness. However, when the system is non-collaborative, it is not possible to avoid idling entirely, since the slowest server has to be idled when the first station is blocked or the second station is starved. Thus, in non-collaborative systems the maximum achievable throughput is reduced due to idling. The following result shows that in two-station, two-server systems with homogeneous

tasks, the loss in the long-run average throughput due to lack of collaboration is bounded and diminishes as the size of the buffer between the stations increases.

Proposition 4. *For a system with two stations, two servers, and homogeneous tasks, the loss in the long-run average throughput due to lack of collaboration is bounded by* $\frac{\mu_2}{2}$*, and converges to zero as* $B_1 \to \infty$*.*

Proof. Let T^*_{col} denote the optimal long-run average throughput for a collaborative system with two stations and two servers. It follows from Theorem 2.1 of Andradóttir et al. [11] that for a system with homogeneous tasks, $T^*_{col} = \frac{\mu_1+\mu_2}{2}$. Thus,

$$\begin{aligned} T^*_{col} - T^{(\delta^{s^*})^\infty} &= \frac{\mu_1+\mu_2}{2} - \frac{\sum_{i=1}^{s^*-1} \mu_1^{B_1+3-s^*+i}\mu_2^{s^*-i} + \sum_{i=0}^{B_1+2-s^*} \mu_1^{B_1+3-i}\mu_2^i}{\sum_{i=1}^{s^*} \mu_1^{B_1+2-i+j}\mu_2^{s^*-i} + \sum_{i=0}^{B_1+2-s^*} \mu_1^{B_1+2-i}\mu_2^i} \\ &= \frac{\mu_1^{B_1+3-s^*}\mu_2^{s^*} + \mu_1^{s^*}\mu_2^{B_1+3-s^*}}{2\Big(\sum_{i=1}^{s^*} \mu_1^{B_1+2-s^*+i}\mu_2^{s^*-i} + \sum_{i=0}^{B_1+2-s^*} \mu_1^{B_1+2-i}\mu_2^i\Big)}. \end{aligned}$$

Note that for a non-collaborative two-station system with homogeneous tasks, and buffer size B_1, the function f that we defined in Section 3.4 becomes

$$f(i) = (\mu_1-\mu_2)\Big(\sum_{j=0}^{B_1+2-i} \mu_1^{B_1+3-i-j}\mu_2^{i+j-1} - \sum_{j=0}^{i-2} \mu_1^{j+1}\mu_2^{B_1+1-j}\Big),$$

and we have

$$f(i) = -f(B_1+4-i) \text{ for } i \in \{1,\dots B+3\}.$$

Thus, the optimal policy uses the threshold $s^* = \frac{B_1+3}{2}$ if B_1 is odd and there are two consecutive optimal threshold points $\lfloor \frac{B_1+3}{2} \rfloor = \frac{B_1+2}{2}$ and $\lceil \frac{B_1+3}{2} \rceil = \frac{B_1+4}{2}$ if B_1 is even, unless $\mu_1 = \mu_2$ and $f(i) = 0,\ \forall i$, in which case $s^* \in \{1,\dots,B_1+2\}$ can be chosen arbitrarily. (Throughout this section, we use $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ to denote the floor and ceiling functions.) Thus, one can choose $s^* = \lfloor \frac{B_1+3}{2} \rfloor$ in all cases, and

$$\begin{aligned} T^*_c - T^{(\delta^{\lfloor \frac{B_1+3}{2} \rfloor})^\infty} &= \frac{\mu_1^{\lceil \frac{B_1+3}{2} \rceil}\mu_2^{\lfloor \frac{B_1+3}{2} \rfloor} + \mu_1^{\lfloor \frac{B_1+3}{2} \rfloor}\mu_2^{\lceil \frac{B_1+3}{2} \rceil}}{2\Big(\sum_{i=1}^{\lfloor \frac{B_1+3}{2} \rfloor} \mu_1^{\lceil \frac{B_1+1}{2} \rceil+i}\mu_2^{\lfloor \frac{B_1+3}{2} \rfloor-i} + \sum_{i=0}^{\lceil \frac{B_1+1}{2} \rceil} \mu_1^{B_1+2-i}\mu_2^i\Big)} \\ &\le \frac{2\mu_1^{\lceil \frac{B_1+3}{2} \rceil}\mu_2^{\lfloor \frac{B_1+3}{2} \rfloor}}{4\mu_1^{B_1+2}} = \frac{\mu_2}{2}\Big(\frac{\mu_2}{\mu_1}\Big)^{\lfloor \frac{B_1+3}{2} \rfloor-1}. \end{aligned}$$

By our assumption that $\mu_1 \geq \mu_2$, $T_c^* - T^{(\delta^{s^*})^\infty}$ is bounded by $\frac{\mu_2}{2}$, and converges to zero as B_1 grows to infinity. $\square$

Proposition 4 shows that if the tasks in a system are similar, the optimal non-collaborative policy will mostly capture the benefits of cross-training if the buffer size is large enough. When the tasks are dissimilar, collaboration might have a more prominent effect on the optimal long-run average throughput. To examine the benefits of collaboration in systems with dissimilar tasks, we consider a system with homogeneous servers and non-homogeneous tasks where the service rates $\mu_{ij} = \gamma_j$ for $i, j \in 1, 2$ depend entirely on the station but not on the server. Without loss off generality, we assume $\gamma_1 \geq \gamma_2$. The following result shows that in such systems, the benefits of collaboration do not completely diminish as the buffer size increases.

Proposition 5. *For a system with two stations, two servers, and homogeneous servers, the loss in the long-run average throughput due to lack of collaboration converges to* $\frac{\gamma_1-\gamma_2}{2}$ *as* $B_1 \to \infty$.

Proof. Similar to the proof of Proposition (4), it follows from Theorem 2.1 of Andradóttir et al.[11] that for a system with homogeneous servers we have $T_{col}^* = \frac{\gamma_1+\gamma_2}{2}$. Since the servers are homogeneous, every non-idling policy yields the same long-run average throughput and

$$T^{(\delta^{s^*})^\infty} = \frac{\sum_{i=0}^{B_1+1} \gamma_2^{i+1}\gamma_1^{B_1+2-i}}{\sum_{i=0}^{B_1+2} \gamma_2^{i}\gamma_1^{B_1+2-i}}$$

for any $s^* \in \{1, \ldots, B_1 + 2\}$, and we have

$$T_{col}^* - T^{(\delta^{s^*})^\infty} = \frac{\gamma_1^{B_1+3} + \gamma_2^{B_1+3}}{2\sum_{i=0}^{B_1+2} \gamma_2^{i}\gamma_1^{B_1+2-i}} = \frac{\gamma_1}{2}\Big(\frac{\gamma_1^{B_1+2}}{\sum_{i=0}^{B_1+2} \gamma_2^{i}\gamma_1^{B_1+2-i}}\Big) + \frac{\gamma_2}{2}\Big(\frac{\gamma_2^{B_1+2}}{\sum_{i=0}^{B_1+2} \gamma_2^{i}\gamma_1^{B_1+2-i}}\Big).$$

Furthermore,

$$\lim_{B_1\to\infty} \frac{\gamma_2^{B_1+2}}{\sum_{i=0}^{B_1+2} \gamma_2^{i}\gamma_1^{B_1+2-i}} = 0,$$
$$\lim_{B_1\to\infty} \frac{\gamma_1^{B_1+2}}{\sum_{i=0}^{B_1+2} \gamma_2^{i}\gamma_1^{B_1+2-i}} = \frac{\gamma_1 - \gamma_2}{\gamma_1}$$

by our assumption that $\gamma_1 \geq \gamma_2$. Thus, the result follows. $\square$

We also numerically compare the collaborative optimal, non-collaborative optimal, best dedicated, and arbitrary (randomly selected) dedicated policies for systems with equal and unequal buffer sizes. We consider systems with $N = \{2, 3, 4, 5\}$ stations, since a numerical study of larger systems has impractically high computational requirements. Tables 3 through 10 show the results when buffers have equal size. In particular, the results for systems with non-homogeneous tasks are provided in Tables 3–6, and the results for systems with homogeneous tasks can be found in Tables 7–10. Our results for systems with unequal buffer sizes are reported in Table 11 for both homogeneous and non-homogeneous tasks.

In all experiments, service times are exponentially distributed and service rates were randomly and independently generated with distribution $U[1, 20]$. Since the computing time is longer for larger systems, the number of replications and the buffer sizes used depend on the system size. We generated 5000 sets of service rates for $N = 2, 3, 4$, and 200 sets of service rates for $N = 5$, both for systems with homogeneous and non-homogeneous tasks. For each set of rates, the long-run average throughputs attained by the policies considered were computed using buffer sizes $B_j \in \{1, 2, \ldots, 10\}$ for $N = 2, 3$ and $B_j \in \{1, 2, \ldots, 5\}$ for $N = 4, 5$. For brevity, we only report the results for buffer sizes $B_j \in \{1, \ldots, 5, 10\}$ for systems with $N = 2, 3$ stations, although the overall averages are computed using the results for all buffer sizes. In all tables, 95% confidence intervals on the long-run average throughput over randomly generated instances are reported. We use the policy iteration algorithm to compute the non-collaborative optimal policy in all settings and the collaborative optimal policy when tasks are non-homogeneous. Note that for collaborative systems with homogeneous tasks, the optimal long-run average throughput is known to be the average of the service rates [11], and hence we do not need to compute a confidence interval on the optimal throughput in this setting. Also, both dedicated server

Table 3: Throughput values for systems with $N = 2$ and non-homogeneous tasks

Buffer	Collaborative Opt.	Non-Collaborative Opt.	Best Ded.	Arbitrary Ded.
1	10.97 ± 0.10	8.97 ± 0.08	8.34 ± 0.09	6.35 ± 0.10
2	11.13 ± 0.10	9.41 ± 0.09	8.78 ± 0.09	6.68 ± 0.10
3	11.23 ± 0.10	9.69 ± 0.09	9.05 ± 0.10	6.85 ± 0.11
4	11.29 ± 0.10	9.87 ± 0.10	9.23 ± 0.10	6.88 ± 0.11
5	11.33 ± 0.10	9.99 ± 0.11	9.36 ± 0.11	6.96 ± 0.11
10	11.41 ± 0.10	10.28 ± 0.11	9.64 ± 0.11	7.23 ± 0.12
Avg. (1-10)	11.29 ± 0.03	9.89 ± 0.03	9.25 ± 0.03	6.96 ± 0.04

Table 4: Throughput values for systems with $N = 3$ and non-homogeneous tasks

Buffer	Collaborative Opt.	Non-Collaborative Opt.	Best Ded.	Arbitrary Ded.
1	12.22 ± 0.07	9.03 ± 0.06	8.21 ± 0.06	4.90 ± 0.08
2	12.46 ± 0.07	9.71 ± 0.06	8.84 ± 0.07	5.19 ± 0.08
3	12.59 ± 0.07	10.12 ± 0.07	9.22 ± 0.07	5.29 ± 0.09
4	$12.67 \pm 0.0.7$	10.40 ± 0.07	9.47 ± 0.08	5.43 ± 0.09
5	12.72 ± 0.08	10.59 ± 0.07	9.65 ± 0.08	5.52 ± 0.09
10	12.83 ± 0.08	10.04 ± 0.08	10.04 ± 0.09	5.57 ± 0.10
Avg. (1-10)	12.67 ± 0.02	10.43 ± 0.02	9.50 ± 0.03	5.42 ± 0.03

assignment policies yield the same throughput in two-station systems when the tasks are homogeneous. Hence, results for only one dedicated server assignment policy are reported in Table 7.

In two-station lines with non-homogeneous tasks and equal buffers, the arbitrary dedicated policy attains only 70% of the non-collaborative optimal throughput. Its performance further worsens for longer lines, yielding 52%, 42%, and 34% of the non-collaborative optimal throughput for systems with three, four and five stations, respectively. However, choosing the best dedicated policy over an arbitrary one results in a considerable improvement, yielding 91% to 95% of the non-collaborative optimal

Table 5: Throughput values for systems with $N = 4$ and non-homogeneous tasks

Buffer	Collaborative Opt.	Non-Collaborative Opt.	Best Ded.	Arbitrary Ded.
1	13.35 ± 0.05	9.25 ± 0.04	8.42 ± 0.05	4.09 ± 0.06
2	13.62 ± 0.05	10.10 ± 0.05	9.19 ± 0.05	4.43 ± 0.07
3	13.76 ± 0.06	10.63 ± 0.05	9.66 ± 0.06	4.43 ± 0.07
4	13.85 ± 0.06	10.98 ± 0.06	9.97 ± 0.06	4.57 ± 0.08
5	13.89 ± 0.06	11.23 ± 0.06	10.20 ± 0.07	4.65 ± 0.08
Avg. (1-5)	13.70 ± 0.02	10.44 ± 0.02	9.49 ± 0.03	4.43 ± 0.03

Table 6: Throughput values for systems with $N = 5$ and non-homogeneous tasks

Buffer	Collaborative Opt.	Non-Collaborative Opt.	Best Ded.	Arbitrary Ded.
1	14.37 ± 0.22	9.32 ± 0.17	8.81 ± 0.18	3.40 ± 0.25
2	14.65 ± 0.22	10.28 ± 0.20	9.73 ± 0.21	3.60 ± 0.27
3	14.79 ± 0.23	10.89 ± 0.23	10.31 ± 0.23	3.71 ± 0.29
4	14.87 ± 0.23	11.31 ± 0.24	10.70 ± 0.25	3.77 ± 0.30
5	14.93 ± 0.23	11.60 ± 0.26	10.98 ± 0.26	3.81 ± 0.30
Avg. (1-5)	14.72 ± 0.10	10.68 ± 0.11	10.11 ± 0.11	3.66 ± 0.13

Table 7: Throughput values for systems with $N = 2$ and homogeneous tasks

Buffer	Collaborative Opt.	Non-Collaborative Opt.	Dedicated
1	10.50	9.11 ± 0.08	6.40 ± 0.10
2	10.50	9.48 ± 0.09	6.71 ± 0.10
3	10.50	9.77 ± 0.09	6.89 ± 0.11
4	10.50	9.93 ± 0.09	7.01 ± 0.11
5	10.50	10.06 ± 0.10	7.09 ± 0.11
10	10.50	10.34 ± 0.10	7.27 ± 0.12
Avg. (1-10)	10.50	9.96 ± 0.03	7.02 ± 0.04

throughput. Our experiments show that allowing servers to collaborate increases the maximum achievable throughput by 14% in systems with two-stations and non-homogeneous tasks. The longer lines benefit from collaboration more, resulting in 21%, 31%, and 38% improvements in systems with three, four and five stations, respectively.

When the tasks are homogeneous, the dedicated policies for two station systems achieve 70% of the non-collaborative optimal throughput, and the performance of the arbitrary dedicated policy for longer lines remain similar to that in systems with non-homogeneous tasks. However, choosing the best dedicated policy yields only a

Table 8: Throughput values for systems with $N = 3$ and homogeneous tasks

Buffer	Collaborative Opt.	Non-Collaborative Opt.	Best Ded.	Arbitrary Ded.
1	10.50	8.78 ± 0.06	5.01 ± 0.08	4.92 ± 0.08
2	10.50	9.34 ± 0.07	5.30 ± 0.09	5.22 ± 0.08
3	10.50	9.69 ± 0.07	5.45 ± 0.09	5.38 ± 0.09
4	10.50	9.91 ± 0.08	5.55 ± 0.09	5.49 ± 0.09
5	10.50	10.06 ± 0.08	5.61 ± 0.10	5.56 ± 0.09
10	10.50	10.36 ± 0.08	5.74 ± 0.10	5.71 ± 0.10
Avg. (1-10)	10.50	9.91 ± 0.02	5.54 ± 0.03	5.49 ± 0.03

Table 9: Throughput values for systems with $N = 4$ and homogeneous tasks

Buffer	Collaborative Opt.	Non-Collaborative Opt.	Best Ded.	Arbitrary Ded.
1	10.50	8.70 ± 0.05	4.23 ± 0.07	4.09 ± 0.06
2	10.50	9.35 ± 0.06	4.46 ± 0.07	4.34 ± 0.07
3	10.50	9.73 ± 0.06	4.58 ± 0.08	4.48 ± 0.08
4	10.50	9.96 ± 0.07	4.65 ± 0.08	4.57 ± 0.08
5	10.50	10.11 ± 0.07	4.69 ± 0.08	4.62 ± 0.08
Avg. (1-5)	10.50	9.57 ± 0.03	4.52 ± 0.03	4.42 ± 0.03

Table 10: Throughput values for systems with $N = 5$ and homogeneous tasks

Buffer	Collaborative Opt.	Non-Collaborative Opt.	Best Ded.	Arbitrary Ded.
1	10.50	8.52 ± 0.20	3.61 ± 0.26	3.54 ± 0.26
2	10.50	9.21 ± 0.22	3.79 ± 0.29	3.61 ± 0.28
3	10.50	9.61 ± 0.24	3.87 ± 0.31	3.81 ± 0.30
4	10.50	9.84 ± 0.25	3.91 ± 0.32	3.81 ± 0.30
5	10.50	9.98 ± 0.26	3.94 ± 0.32	3.87 ± 0.30
Avg. (1-5)	10.50	9.43 ± 0.08	3.82 ± 0.07	3.73 ± 0.10

slight improvement achieving 56%, 47%, and 41% of the non-collaborative optimal throughput for systems with three, four, and five stations, respectively. The longer lines benefit from collaboration more as in the systems with non-homogeneous tasks, however the overall improvement achieved is comparatively lower with 5% to 11% increase in the long-run average throughput.

In the experiments for systems with unequal buffer sizes, the service rate data from the aforementioned experiments were used. Note that there is a single buffer in two-station systems, thus we report results for $N = 3, 4, 5$ only. For each set of rates, we randomly generated ten sets of buffer sizes for systems with $N = 3$ and five sets of buffer sizes for systems with $N = 4, 5$ from the discrete uniform distributions $U\{1, 2, , \ldots, 10\}$ and $U\{1, 2, \ldots, 5\}$, respectively. For each instance of service rates, the buffer size values generated at different stations are independent of each other. Our results show that allowing different sizes for each buffer in a system does not significantly affect the overall average gap between the optimal policies in collaborative and non-collaborative systems or the average performance of dedicated policies. In fact, the results shown in Table 11 are similar to the aggregated results

Table 11: Throughput values for systems with $N = 3, 4, 5$ and unequal buffer sizes

Non-Homogeneous Tasks				
N	Collaborative Opt.	Non-Collaborative Opt.	Best Ded.	Arbitrary Ded.
3	12.68 ± 0.02	10.42 ± 0.02	9.48 ± 0.03	5.43 ± 0.03
4	13.71 ± 0.02	10.41 ± 0.02	9.48 ± 0.03	4.40 ± 0.03
5	14.73 ± 0.10	10.61 ± 0.10	10.08 ± 0.10	3.86 ± 0.14
Homogeneous Tasks				
N	Collaborative Opt.	Non-Collaborative Opt.	Best Ded.	Arbitrary Ded.
3	10.50	9.92 ± 0.02	5.58 ± 0.03	5.48 ± 0.03
4	10.50	9.58 ± 0.03	4.56 ± 0.03	4.42 ± 0.03
5	10.50	9.43 ± 0.12	4.01 ± 0.14	3.86 ± 0.13

reported in Tables 4 through 6 and 8 through 10.

Cross-training can improve the maximum achievable throughput through dynamic server allocation and collaboration, in systems with both homogeneous and non-homogeneous tasks. However, our results show that the benefits gained through these mechanism are dependent on similarity or dissimilarity of the tasks in the system. Collaboration improves the maximum attainable throughput more when the tasks are non-homogeneous. In a system with non-homogeneous tasks, the performance of a server may differ at different stations, and allowing collaboration makes it possible to utilize each server where they perform better more often, consequently increasing the long-run average throughput. We also observe that the improvement in the long-run average throughput due to collaboration decreases as the buffer size increases and that non-collaborative systems are more sensitive to buffer size. These results are consistent with Propositions 4 and 5, and can be explained by the observation that collaboration is more valuable when blocking is more frequent, since it allows all servers to be utilized even when some stations are blocked.

When tasks are homogeneous, all dedicated policies perform poorly. On the other hand, choosing the best dedicated policy can greatly improve the throughput achieved if the tasks are non-homogeneous. If the service rates depend on the task, it is possible that at each station there is a distinct server that performs the best (i.e., there are

no dominated servers), in which case a dedicated server assignment policy is optimal (see Theorem 1). On the other hand, if the tasks are homogeneous then the servers dominate each other in the sense that they can be ordered with respect to service rates. In this case, it is beneficial to utilize the fastest server in multiple stations, and substantial improvement can be gained through dynamic assignment of the servers.

As the number of stations in the system grows, the chance of blocking increases, causing the non-collaborative optimal throughput to decrease when tasks are homogeneous. In collaborative systems, this is counteracted since any blockage can be quickly resolved. However, if the tasks are non-homogeneous, both the collaborative and non-collaborative optimal throughputs improve with growing system size, since it is more likely to have a faster server available at each station.

3.6 Server Assignment Heuristics for Non-Collaborative Lines with $N \leq 3$

In this section, we develop four different server assignment heuristics for N-station, N-server non-collaborative systems. Each heuristic uses a primary assignment to allocate servers to stations, and changes the allocation dynamically according to a contingency plan when prospects of blocking or starving are high, as motivated by the behavior of the optimal policy in systems with two stations. We provide a numerical study that compares the long-run average throughput of the heuristics to that of the non-collaborative optimal policy for both equal and unequal buffer sizes. The best dedicated policy and an arbitrary (randomly selected) dedicated policy were also used as benchmark policies in the comparison.

Our results in Section 3.5 show that when tasks are non-homogeneous, the best dedicated server assignment policy yields near-optimal throughputs. However, when the tasks are homogeneous, an efficient server allocation strategy is unknown for $N \geq 3$. Thus, in this section, we focus on systems with homogeneous tasks and assume the service rates do not depend on the station but only on the server, i.e.,

$\mu_{ij} = \mu_i$ for $i \in \{1, \ldots, N\}$, $j \in \{1, \ldots, N\}$. In Section 3.4, we identified the optimal server allocation policy for the two-station, two-server system with an arbitrary buffer size. The optimal policy is of threshold type with a threshold value s^* and it does not involve primary assignment of servers to stations. Bartholdi and Eisenstein [19] have shown that in bucket brigade manufacturing, the production line balances itself and attains the maximum possible throughput when servers are ordered from slower to faster. On the other hand, Hillier and Boling [38], have demonstrated and analyzed the so-called bowl phenomenon that shows that the throughput of a production system may be increased by deliberately unbalancing the line and observed that the bowl phenomenon occurs when the greatest amount of work is allocated to the stations on the ends of the line and the smallest amounts should be allocated to the stations in the middle. We use these ideas to devise the primary assignments for our heuristics and use our results from Section 3.4 to develop the accompanying contingency plan.

More specifically, we consider four primary assignments. Our primary assignments are defined as follows: Let us number the servers so that we have $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_N$. Then the first primary assignment will be *slower-to-faster* (S-to-F) which assigns server N to the first station, server $N-1$ to the second station, and so on (as in bucket brigades). Another primary assignment will be *faster-to-slower* (F-to-S) which assigns server 1 to the first station, server 2 to the second station, and so on. The third primary assignment is the *bowl* assignment which assigns server 1 to the middle station ($\lceil \frac{N+1}{2} \rceil$), server 2 to the left of server 1, server 3 to the right of server 1, server 4 to the left of server 2, server 5 to the right of server 3, and so on. Note that the fastest servers are assigned to the middle stations and the slowest servers are assigned to the end stations. For our fourth heuristic, we use an *arbitrary* (randomly selected) primary assignment. For all heuristics, if there are k starved or blocked stations, the N-station system is reduced to an $(N-k)$-station system and the primary assignment

uses the $N-k$ fastest servers.

Our contingency plan locally improves portions of the N-station line with the purpose of preventing stations from being starved or blocked. In order to dynamically change the server assignment, it uses $N-1$ threshold values $s_1^*, s_2^*, \ldots, s_{N-1}^*$. These thresholds are calculated as in Section 3.4 for subsystems that consist of two consecutive stations of our original N-station system with servers assigned to these stations following the specified primary assignment. In particular, we let $s_j^*, j \in \{1, \ldots N-1\}$, denote threshold point of a two-station system with a buffer of size B_j and with the servers that are assigned to station j and station $j+1$ in the N-station system. Note that for a two-station system with homogeneous tasks ($\mu_{ij} = \mu_i$), and buffer size B, we can assign $s_j^* = \lfloor \frac{B_j+3}{2} \rfloor$as shown in the proof of Proposition 4.

Using the threshold values s_j^*, one can construct the heuristic policies as follows. We start with the chosen primary assignment in each state. For the *slower-to-faster* heuristic, we keep the primary assignment if $s_j \geq s_j^*$ for all j, and reverse the assignments of servers in each sequence of stations $j, \ldots, j+m$ such that $s_j < s_j^*, \ldots, s_{j+m} < s_{j+m}^*$, where j is as small as possible and m is as large as possible, so that the server assigned to station j will be assigned to station $j+m+1$, the server assigned to station $j+1$ will be assigned to station $j+m$ and so on (i.e., faster servers move *backwards*). Similarly, for *faster-to-slower* heuristic, we keep the primary assignment if $s_j < s_j^*$ for all j, and reverse the assignments of servers in each sequence $j, \ldots, j+m$ such that $s_j \geq s_j^*, \ldots, s_{j+m} \geq s_{j+m}^*$, where j is as small as possible and m is as large as possible (i.e., faster servers move *forward*).

For the *bowl* heuristic:

- If we have

$$s_j \geq s_j^* \text{ for all } j \leq N/2 \tag{7}$$

and

$$s_j < s_j^* \text{ for all } j > N/2, \tag{8}$$

then we keep the primary assignment.

- If there is a sequence $j, j+1, j+2, \ldots, j+m$ with $j > N/2$ or $j+m \leq N/2$ that does not satisfy the conditions given in (7) and (8), where j is as small as possible and m is as large as possible, we reverse the assignments of the servers in the sequence.

- If there is a sequence $j, j+1, j+2, \ldots, j+m$ with $j \leq N/2$ and $j+m > N/2$ that does not satisfy the conditions given in (7) and (8), where j is as small as possible and m is as large as possible, we reverse the assignments of the servers at both stations j to $\lfloor N/2 \rfloor$ and stations $\lceil \frac{N+1}{2} \rceil$ to $j+m+1$ (so that the servers that were originally assigned to stations $j, \ldots, \lfloor N/2 \rfloor$ now will be assigned to stations $\lfloor N/2 \rfloor, \ldots, j$, respectively and the servers that were originally assigned to stations $\lceil \frac{N+1}{2} \rceil, \ldots, j+m+1$ now will be assigned to stations $j+m+1 \ldots, \lceil \frac{N+1}{2} \rceil$, respectively).

For an *arbitrary* primary assignment, similar logic applies, so that if there is a sequence of stations so that in each consecutive pair of stations the faster server needs to move *backwards* (e.g., there is a sequence of stations $j, j+1, \ldots, j+m$ such that $s_j < s_j^*, \ldots, s_{j+m} < s_{j+m}^*$ and the servers at these stations are ordered from slower to faster) or *forward* (e.g., there is a sequence of stations $j, j+1, \ldots, j+m$ such that $s_j \geq s_j^*, \ldots, s_{j+m} \geq s_{j+m}^*$ and the servers at these stations are ordered from faster to slower), we reverse the server assignments for that portion of the line.

The heuristics are evaluated through numerical experiments. We consider systems with $N \in \{3, 4, 5\}$ stations, since a numerical study of larger systems has impractically high computational requirements. We use the same experimental setup and service time data as in our previous experiments for systems with homogenous tasks in Section 3.5. For each setting, we compare the long-run average throughputs attained by the optimal non-collaborative policy, our four heuristic policies, and the

Table 12: Throughput values for systems with $N = 3$ and equal buffer sizes

B.	Non-C. Opt.	S-to-F	F-to-S	Bowl	Arbitrary	Best D.	Arbitrary D.
1	8.78 ± 0.06	8.68 ± 0.06	8.65 ± 0.06	8.62 ± 0.06	8.52 ± 0.06	5.01 ± 0.08	4.92 ± 0.08
2	9.34 ± 0.07	9.23 ± 0.07	9.12 ± 0.07	9.13 ± 0.07	9.04 ± 0.07	5.30 ± 0.09	5.22 ± 0.08
3	9.69 ± 0.07	9.51 ± 0.07	9.34 ± 0.07	9.39 ± 0.07	9.32 ± 0.07	5.45 ± 0.09	5.38 ± 0.09
4	9.91 ± 0.08	9.66 ± 0.08	9.47 ± 0.07	9.54 ± 0.07	9.48 ± 0.07	5.55 ± 0.09	5.49 ± 0.09
5	10.06 ± 0.08	9.76 ± 0.08	9.54 ± 0.08	9.63 ± 0.08	9.57 ± 0.08	5.61 ± 0.10	5.56 ± 0.09
10	10.36 ± 0.08	9.94 ± 0.08	9.68 ± 0.08	9.79 ± 0.08	9.74 ± 0.08	5.74 ± 0.10	5.71 ± 0.10
Avg.	9.91 ± 0.02	9.63 ± 0.02	9.44 ± 0.02	9.50 ± 0.02	9.44 ± 0.02	5.54 ± 0.03	5.49 ± 0.03

Table 13: Throughput values for systems with $N = 4$ and equal buffer sizes

B.	Non-C. Opt.	S-to-F	F-to-S	Bowl	Arbitrary	Best D.	Arbitrary D.
1	8.70 ± 0.05	8.65 ± 0.05	8.59 ± 0.05	8.57 ± 0.05	8.54 ± 0.05	4.23 ± 0.07	4.09 ± 0.06
2	9.35 ± 0.06	9.25 ± 0.06	9.04 ± 0.06	9.10 ± 0.06	9.05 ± 0.06	4.46 ± 0.07	4.34 ± 0.07
3	9.73 ± 0.06	9.56 ± 0.06	9.26 ± 0.06	9.38 ± 0.06	9.31 ± 0.06	4.58 ± 0.08	4.48 ± 0.08
4	9.96 ± 0.07	9.73 ± 0.06	9.37 ± 0.06	9.53 ± 0.06	9.47 ± 0.06	4.65 ± 0.08	4.57 ± 0.08
5	10.11 ± 0.07	9.84 ± 0.07	9.44 ± 0.06	9.63 ± 0.07	9.57 ± 0.07	4.69 ± 0.08	4.62 ± 0.08
Avg.	9.57 ± 0.03	9.41 ± 0.03	9.14 ± 0.03	9.24 ± 0.03	9.19 ± 0.03	4.52 ± 0.03	4.42 ± 0.03

two dedicated benchmark policies. Tables 12 through 14 show our results for systems with equal buffer sizes for $N = 3, 4, 5$, respectively, and Table 15 shows the results for systems with unequal buffers for $N = 3, 4, 5$. All tables display 95% confidence intervals.

Tables 12, 13, and 14 show that all heuristics yield near-optimal results when the system size is relatively small. Hence we conclude that all primary assignments we consider work well when accompanied by the contingency plan. Moreover, the *slower-to-faster* heuristic performs better than the remaining heuristics for all systems and buffer sizes, with the exception of systems with five stations and unit buffer size. For small buffer sizes, *faster-to-slower* heuristic outperforms the *bowl* heuristic

Table 14: Throughput values for systems with $N = 5$ and equal buffer sizes

B.	Non-C. Opt.	S-to-F	F-to-S	Bowl	Arbitrary	Best D.	Arbitrary D.
1	8.52 ± 0.20	7.86 ± 0.19	7.95 ± 0.19	8.04 ± 0.20	7.79 ± 0.19	3.61 ± 0.26	3.54 ± 0.26
2	9.21 ± 0.22	8.53 ± 0.21	8.23 ± 0.22	8.48 ± 0.22	8.29 ± 0.22	3.79 ± 0.29	3.61 ± 0.28
3	9.61 ± 0.24	8.97 ± 0.23	8.35 ± 0.23	8.69 ± 0.23	8.49 ± 0.23	3.87 ± 0.31	3.81 ± 0.30
4	9.84 ± 0.25	9.27 ± 0.24	8.42 ± 0.24	8.81 ± 0.24	8.77 ± 0.24	3.91 ± 0.32	3.81 ± 0.30
5	9.98 ± 0.26	9.46 ± 0.25	8.45 ± 0.24	8.87 ± 0.25	8.72 ± 0.25	3.94 ± 0.32	3.87 ± 0.31
Avg.	9.43 ± 0.08	8.82 ± 0.08	8.28 ± 0.07	8.58 ± 0.07	8.41 ± 0.07	3.82 ± 0.10	3.73 ± 0.09

and the heuristic with an arbitrary primary assignment, and it is outperformed by these heuristics for larger values of buffer size. In general, we observe a large gap between the performance of the heuristic policies and the performance of dedicated assignments. Furthermore, as the number of stations in the system increases, the performance of the heuristic policies improves relative to that of dedicated policies since the throughput values attained by dedicated policies rapidly deteriorates. The decrease in the performance of our heuristics as the number of stations increases is expected since our contingency plan only aims to improve the system performance locally.

In particular, for the three-station systems, the throughput attained by the *slower-to-faster* heuristic was approximately 97% of the throughput attained by the optimal policy when the throughput values are averaged over different buffer sizes (98% when averaged over buffer sizes 1 through 5; note that for larger systems a smaller range of buffer sizes is used). By contrast, the throughput attained by the dedicated polices remained around 56% of the maximum throughput. For systems with four and five stations, the *slower-to-faster* heuristic attained 98% and 91% of the optimal throughput, whereas the throughput attained by the dedicated polices remained around 47% and 41% of the optimal, respectively.

Table 15 shows the results when the buffers between different stations are allowed to have different sizes. In this experiment, we used the same service rate data as in Tables 12, 13, and 14. For each set of rates, different sets of buffer sizes were randomly generated, as was done in Section 3.5 for Table 11. Consistent with Section 3.5, our experiments show that allowing the buffer sizes to differ does not affect the performance of our heuristics significantly. For systems with unequal buffer sizes, the *slower-to-faster* heuristic on average achieved approximately 97%, 98%, and 93% of the maximum long-run average throughput for $N = 3, 4, 5$, respectively (recall that a larger range of buffer sizes was considered for $N = 3$). The corresponding best

Table 15: Throughput values for systems with $N = 3, 4, 5$ and unequal buffer sizes

N	Non-C. Opt.	S-to-F	F-to-S	Bowl	Arbitrary	Best D.	Arbitrary D.
3	9.92 ± 0.02	9.63 ± 0.02	9.44 ± 0.02	9.50 ± 0.02	9.37 ± 0.02	5.58 ± 0.03	5.48 ± 0.03
4	9.58 ± 0.03	9.40 ± 0.03	9.14 ± 0.03	9.24 ± 0.03	9.18 ± 0.03	4.56 ± 0.03	4.42 ± 0.03
5	9.43 ± 0.12	8.76 ± 0.11	8.30 ± 0.11	8.58 ± 0.11	8.35 ± 0.11	4.01 ± 0.14	3.86 ± 0.13

dedicated policy on average achieved 56%, 48%, and 43% of the maximum long-run average throughput for $N = 3, 4, 5$, respectively. In fact, averaging over different buffer sizes in Tables 12, 13, 14 gives us similar results.

It is already known that ordering the servers from slower to faster yields the largest throughput in bucket brigade manufacturing where there is no blocking [19]. This allocation strategy cannot be optimal in the system we study, since even the best dedicated server assignment policy is suboptimal. However, we have shown that the *slower-to-faster* primary assignment is near-optimal even when there is blocking, as long as the order of the servers is allowed to change according to our contingency plan. Hence our heuristic provides an efficient workforce management strategy when the optimal policy is hard to identify. Furthermore, all of the dynamic server assignment heuristics we developed yield substantially higher long-run average throughputs than the best dedicated assignment, implying that our contingency plan greatly contributes to the long-run average throughput.

3.7 Conclusions

We study systems of tandem queues with flexible, non-collaborative servers and finite buffers between the stations. We prove when a distinct server is the fastest at each station, the optimal policy keeps the servers dedicated at the stations where they are the fastest, so cross-training is not beneficial. Otherwise, a more complicated server allocation policy is needed as the servers should be utilized at multiple stations. Furthermore, we completely characterize the server assignment policy that maximizes the long-run average throughput for Markovian two-station, two-server systems. For

systems with two stations, numerical results showed that the structure of the optimal policy is insensitive to the distribution of the service requirements, and the optimal policy in Markovian systems can be used as a near-optimal heuristic in non-Markovian systems.For larger Markovian systems with homogeneous tasks, we develop server allocation heuristics that are based on our results for the two-station systems. Each heuristic is composed of a primary server assignment and a contingency plan that dynamically reallocates the servers using multiple thresholds. Numerical results show that all primary assignments we consider yield near-optimal results when accompanied by our contingency plan. Particularly, ordering the servers from slower to faster yields the best results. These results suggest multiple threshold type policies may be near-optimal for systems with more than two stations.

We also provide a comparison of collaborative, non-collaborative, and non-flexible (dedicated) systems.For systems with two stations and homogeneous tasks, we show that the benefits of collaboration diminish as buffer size increases, and dynamic server allocation can compensate for the lack of collaboration. On the other hand, if the tasks are non-homogeneous, collaboration can remain considerably beneficial. Numerical results for larger systems also suggest that collaboration improves the long-run average throughput more when tasks are non-homogeneous, whereas cross-training improves system performance mostly through dynamic server allocation in systems with homogeneous tasks. Our results show that server flexibility is useful even if the servers are non-collaborative, and dynamic server allocation can increase the maximum long-run average throughput without a bound as the differences between the skill levels of the servers become larger.

CHAPTER IV

NON-COLLABORATIVE TANDEM NETWORKS WITH SETUP COSTS

4.1 Introduction

In this section, we study the effects of setups on the optimal server allocation policy in systems with cross-trained and non-collaborative servers, and characterize the conditions under which cross-training is still advantageous.

Consider a tandem queueing system with N stations, N servers, and finite buffers between the stations. We assume that the system operates under the manufacturing blocking mechanism, so a station becomes blocked whenever the buffer following it is full and a job completion occurs at the station. We also assume that there is an infinite supply of jobs in front of the first station and infinite space for completed jobs after the last station. Service requirements at each station are independent and identically distributed. Servers are cross-trained to work at different stations, and a server can move from one station to another after each service completion. However, the system is *non-collaborative* so that servers are not allowed to work together at a station at any time. Since the number of servers and stations are equal, this implies that there is always a single server at each station. We let $\mu_{ij} \geq 0$ denote the rate of server i at station j, and let $0 \leq B_j < \infty$ for $j \in \{1, \ldots, N-1\}$ denote the size of the buffer between stations j and $j+1$. We assume, without loss of generality, that the mean service requirement is 1 at all stations.

We assume that travel and setup times are negligible, but (non-negative) setup costs are incurred whenever the servers are reassigned. Moreover, each departure from the system results in a revenue of r. Without loss of generality, we let $r = 1$.

Our objective in this chapter is to identify the dynamic server assignment policy that maximizes the long-run average profit for the system described.

The remainder of this chapter is organized as follows. In Section 4.2, we give a result for the N-station system with general service requirements and structured service rates. In Section 4.3 we formulate the optimization problem and give an equivalent Markov decision process formulation. The optimal policy for the special case of homogeneous tasks and costs is completely characterized in Section 4.4. We develop heuristic server assignment policies for systems with non-homogeneous tasks and/or non-constant setup costs in Section 4.5, and provide a numerical study that evaluates the performance of the heuristics. Our conclusions are summarized in Section 4.6. Finally, additional numerical results are provided in the Appendices A and B.

4.2 Systems with General Service Rates

In Chapter 3, we showed that for a system of N stations and N servers without setups, if for each station there exists one distinct server that is better than all other servers at that station, then the dedicated server assignment policy that keeps each server where they are the fastest is optimal. The following proposition generalizes that result for systems with positive setup costs.

Proposition 6. *Consider an N station tandem line with N non-collaborative servers and service requirements with general distributions for which a server reassignment results in setup costs. Suppose there exists $i_1, \ldots, i_N$ with $\{i_1, i_2, \ldots, i_N\} = \{1, 2, \ldots, N\}$ such that $\mu_{i_1 1} \geq \max\{\mu_{i_2 1}, \mu_{i_3 1}, \ldots, \mu_{i_N 1}\}$, $\ldots$, $\mu_{i_N N} \geq \max\{\mu_{i_1 N}, \mu_{i_2 N}, \ldots, \mu_{i_{N-1} N}\}$. Then the policy that assigns server i_k to station k at all times for all $k \in \{1, 2, \ldots, N\}$ is optimal.*

Proof. If for each station, there is a distinct server that is better than all other other servers at that station, then (by Theorem 1 in Chapter 3) any server assignment policy π' that is different from the policy described in Proposition 6, will achieve

lesser throughput. Furthermore if setup costs are present, any policy that is not dedicated may incur setup costs, resulting in a lower long-run average profit. Hence, the policy given in Proposition 6 is optimal. □

Remark 4. *Note that if a subset of servers $I \subset \{1, \ldots, N\}$ are specialized at stations $J = \{j_i : i \in I\} \subset \{1, \ldots, N\}$ but dominated at stations $j \notin J$ by servers $i \notin I$, then the suboptimality of policies that move a server $i \in I$ away from station j_i can be proved in similar manner (see Proposition 2 in Chapter 3).*

For all server assignment policies π and $t \geq 0$, let $D_\pi(t)$ be the number of departures from the system and $C_\pi(t)$ be the cumulative setup cost incurred under the server assignment policy π in the period $[0, t]$. Let P_π be the long-run average profit under policy π. For any system with finite setup costs and service requirements, we have

$$P_\pi = \lim_{t\to\infty} \mathbb{E}\left\{ \frac{D_\pi(t)}{t} - \frac{C_\pi(t)}{t} \right\}. \tag{9}$$

Similarly, for all server assignment policies π and $t \geq 0$, define $K_\pi(t)$ be the number of server reassignments occurred under the server assignment policy π up to time t. We consider any policy π such that $\lim_{t\to\infty} \frac{K_\pi(t)}{t} = 0$ to be dedicated. Proposition 6 states that when there are setup costs, if it is not possible to increase the service rates at any of the stations by moving the servers, a dedicated policy is optimal. Our next result shows that for general N-station, N-server systems, a dedicated policy maximizes the long-run average profit if the setup costs are sufficiently large.

Proposition 7. *For an N-station tandem line with N non-collaborative servers and service requirements with general distributions, the optimal policy becomes dedicated as all setup costs tend to infinity.*

Proof. Note that the first term of the limit in (9) is independent of the magnitude of the setup costs. Let the largest service rate available in the system be denoted by

μ. Then, the term $\lim_{t\to\infty} \mathbb{E}\left\{\frac{D_\pi(t)}{t}\right\}$ is also bounded above by μ since the servers are non-collaborative. Moreover, the second term grows as the setup costs increase for any non-dedicated policy. Let the set of such policies be denoted by Π'. For any policy $\pi \in \Pi'$, there exists a setup cost value $c'_\pi < \infty$, such that for all $c > c'_\pi$ the long-run average costs $\lim_{t\to\infty} \mathbb{E}\left\{\frac{C_\pi(t)}{t}\right\} \geq \mu$ and the long-run average profit $P_\pi \leq 0$. Thus, all non-dedicated policies result in negative long-run average profits as the setup costs increase.

For a dedicated server assignment policy, the long-run average setup cost incurred is zero. Hence, the long run average profit achieved by such a policy is equal to the long-run average throughput (i.e., it is constant and nonnegative) for all values of setup costs. Therefore, a dedicated server a assignment policy becomes optimal as setup costs tend to infinity. □

4.3 Problem Formulation

We formulate the dynamic server assignment problem for a tandem system with N stations and N non-collaborative servers in the presence of constant setup costs $c \geq 0$ that are independent of the location of the servers. Assuming that the service requirements are exponentially distributed, we translate the continuous time problem into a discrete time Markov decision problem.

We define the stochastic process $\{Y^\pi(t) : t \geq 0\}$ with state space S_Y for a policy $\pi \in \Pi$, where Π is the set of server assignment policies under consideration, as follows: $Y^\pi(t) = (Y_1^\pi(t), \ldots, Y_{N-1}^\pi(t))$, where $Y_j^\pi(t) \in \{0, \ldots, B_j + 2\}$ is the number of jobs that have been processed at the first j stations but not at station $j+1$ at time t for $j \in \{1, \ldots, N-1\}$,. Similarly, for all server assignment policies $\pi \in \Pi$, we let $Z^\pi(t) = (Z_1^\pi(t), \ldots, Z_N^\pi(t))$ for all $t \geq 0$, where $Z_i^\pi(t), i \in \{1, \ldots, N\}$ denotes the station that server i was assigned to under the policy π at the time of the most recent service completion prior to time t, and let S_Z be the state space of the stochastic process

$\{Z^\pi(t)\}$. We use the stochastic process $\{X^\pi(t)\}$, where $X^\pi(t) = (Y^\pi(t), Z^\pi(t))$ for all $t \geq 0$, to model the state of the system under policy π as a function of time.

Let Π denote the set of all Markovian stationary deterministic server assignment policies that correspond to the state space S of the stochastic processes $\{X^\pi(t)\}$. In other words, the policies in Π specify the server assignments as a function of the current state $x = (y, z) \in S$ of the stochastic process $\{X(t)\}$, where $y = (s_1, \ldots, s_{N-1}) \in S_Y$ and $z = (z_1, \ldots, z_N) \in S_Z$. Hence, the server assignments may depend on the status of the stations and the buffers, as well as the previous locations of the servers.

For all $x \in S$, let A_x denote the allowable actions in state x. We use the notation $a_{\sigma_1\sigma_2\ldots\sigma_N}$ to represent the actions, where $\sigma_i = (\sigma_{i,1}, \sigma_{i,2})$ and $\sigma_{i,1}$ is the station server i is assigned to with the convention that $\sigma_{i,2} = 0$ if server i is voluntarily idled and $\sigma_{i,2} = 1$ otherwise. Thus, we have $A_x = A = \cup_{\sigma \in (\{1,\ldots,N\}\times\{0,1\})^N} \{a_\sigma\}$ for all $x \in S$. We let d denote a decision rule where $d(x) \in A_x$ for all $x \in S$. Hence the policy $\pi \in \Pi$ that corresponds to the decision rule d can be represented as $\pi = (d)^\infty$.

Since the state space of our Markov chain $\{X^\pi(t)\}$, as well as the immediate rewards, are finite, the limit given in equation (9) exists by Proposition 8.1.1 of Puterman [56]. We are interested in the following optimization problem:

$$\max_{\pi \in \Pi} P_\pi. \tag{10}$$

Let π_a be the server assignment policy that chooses action a in every state $x \in S$. Note that the stochastic process $\{Y^{\pi_a}(t)\}$, $\pi_a \in \Pi$ is a continuous time Markov chain, and let $Q_a(y, y')$, $a \in A$ be the rate at which the continuous time Markov chain $\{Y^{\pi_a}(t)\}$ goes from state y to state y'. Then the stochastic process $\{X^\pi(t)\}$ is a continuous time Markov chain for all $\pi = (d)^\infty \in \Pi$, with transition rates

$$q_d(x, x') = \begin{cases} Q_{d(x)}(y, y') \text{ if } z' = Z_{d(x)}, \\ 0 \text{ otherwise,} \end{cases}$$

for all $x = (y, z), x' = (y', z') \in S$, where $Z_{d(x)} = \{z_{d,1}(x), \ldots, z_{d,N}(x)\}$ and $z_{d,i}(x) = \sigma_{i,1}$ when $d(x) = a_{\sigma_1\sigma_2\ldots\sigma_N}$.

It is clear that there exists a finite uniformization constant $q \geq \sum_i \max_j \mu_{ij}$. Hence, the process $\{X^\pi(t)\}$ is uniformizable for all $\pi \in \Pi$, and the continuous time optimization problem can be translated into an equivalent discrete time Markov decision problem. For all $y = (s_1, s_2 \ldots, s_{N-1}) \in S_Y$, let $D_y = \varnothing$ if $s_{N-1} = 0$, and $D_y = \{(s_1, s_2 \ldots, s_{N-1} - 1)\}$ if $s_{N-1} > 0$. Also, let $\mathbb{1}_{\{.\}}$ be the indicator function and define summation over an empty set to be zero. Andradóttir et al. [14] show that the original problem (10) is equivalent to

$$\max_{\pi\in\Pi} \lim_{K\to\infty} \mathbb{E}\Big\{\frac{1}{K}\sum_{k=1}^{K} R_d(X'_\pi(k-1))\Big\}, \tag{11}$$

where $X'_\pi(k)$ is the discrete time Markov chain that corresponds to the original process $X_\pi(t)$ and

$$R_d(x) = \sum_{y'\in D_y} Q_{d(x)}(y, y') - \Big(\sum_{y'\in S_Y\setminus\{y\}} Q_{d(x)}(y, y')\Big) \times \Big(\sum_{i=1}^{N} c\mathbb{1}_{\{z_i\neq z_{d,i}(x)\}}\Big)$$

is the reward received when the decision rule d is used in state x. In the remainder of this chapter, we work with the alternative formulation (11).

4.4 Systems with Homogeneous Tasks

In this section, we consider the two-station system with homogeneous tasks, and completely characterize the optimal policy in the presence of constant setup costs. Since the tasks are homogeneous, the service rates are only dependent on the server, and we have $\mu_{ij} = \mu_i$ for $i \in \{1, 2\}$. Throughout our developments, without loss of generality we assume that $\mu_1 \geq \mu_2$, so that server 1 works faster than server 2 at both stations. Furthermore, if $\mu_1 = \mu_2$ then the servers are identical and any dedicated server assignment policy that avoids setup costs is optimal (see Proposition 6). Hence, without loss of generality, we can focus on the case where server 1 is strictly faster

than server 2 (i.e., $\mu_1 > \mu_2$). We use $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ to denote floor and ceiling functions, and define the set of decision rules δ_k for $k \in 0, \ldots, \lfloor\frac{B_1+3}{2}\rfloor$ as follows:

$$\delta_k(x) = \begin{cases} a_{(1,1)(2,1)} & \text{if } x = (s,2,1), s < k, \\ a_{(2,1)(1,1)} & \text{if } x = (s,2,1), s \geq k, \\ a_{(1,1)(2,1)} & \text{if } x = (s,1,2), s \leq B_1 + 2 - k, \\ a_{(2,1)(1,1)} & \text{if } x = (s,1,2), s > B_1 + 2 - k. \end{cases}$$

The policy $(\delta_k)^\infty$ is a non-idling double threshold policy with symmetric thresholds that depend on both the number of jobs that are in the buffer and the server assignment prior to the last job completion.

Before we present the result that completely characterizes the optimal server assignment policy for arbitrary buffer size B_1, we define the following bounds c_k, $k \in \{1, \ldots, \lfloor\frac{B_1+3}{2}\rfloor\}$ on the setup cost c:

$$c_k = \frac{\sum_{i=0}^{B_1+3-2k} \sum_{j=0}^{i} \mu_1^j \mu_2^{i-j} (\mu_1^{B_1+3-k-i} \mu_2^{k-1} - \mu_1^{k-1} \mu_2^{B_1+3-k-i})}{4\Big(\sum_{i=0}^{k-2} \mu_2^i \mu_1^{B_1+2-i} + \sum_{i=0}^{B_1+3-k} \mu_2^i \mu_1^{B_1+2-i} \Big)}, \tag{12}$$

and show that $c_{k+1} < c_k$ for all $k \in \{1, \ldots, \lfloor\frac{B_1+3}{2}\rfloor - 1\}$.

Lemma 2. *The bounds c_k, $k \in \{1, \ldots, \lfloor\frac{B_1+3}{2}\rfloor\}$ are decreasing in k.*

Proof. Let c'_k denote the numerator of c_k defined in (12). We have

$$\begin{aligned} c'_k - c'_{k+1} = & \sum_{i=0}^{B_1+1-2k} \sum_{j=0}^{i} \mu_1^j \mu_2^{i-j} (\mu_1^{B_1+2-k-i} \mu_2^{k-1} (\mu_1 - \mu_2) + \mu_1^{k-1} \mu_2^{B_1+2-k-i} (\mu_1 - \mu_2)) \\ & + \sum_{i=B_1+2-2k}^{B_1+3-2k} \sum_{j=0}^{i} \mu_1^j \mu_2^{i-j} (\mu_1^{B_1+3-k-i} \mu_2^{k-1} - \mu_1^{k-1} \mu_2^{B_1+3-k-i}) \\ > & \, 0, \end{aligned}$$

for $k \in \{1, \ldots, \lfloor\frac{B_1+3}{2}\rfloor - 1\}$ due to our assumption that $\mu_1 > \mu_2$. Thus, the numerator is decreasing in k. Also let c''_k denote the denominator in (12). Similarly, we have

$$c''_k - c''_{k+1} = 4(\mu_2^{B_1+3-k} \mu_1^{k-1} - \mu_2^{k-1} \mu_1^{B_1+3-k}) < 0,$$

and the denominator is decreasing in k, finishing the proof. □

Theorem 3. *For a two-station Markovian tandem line with flexible but non-collaborative servers, if the tasks are homogeneous and server 1 is strictly faster than server 2, i.e., $\mu_{1j} = \mu_1 > \mu_2 = \mu_{2j}$ for all $j \in \{1, 2\}$, then the optimal server assignment policy $\pi^* = (\delta^*)^\infty$ is as follows:*

(i) If $c \geq c_1$, then $\delta^* = \delta_0$, and the optimal server assignment policy is dedicated.

(ii) If $c_{k+1} \leq c \leq c_k$, then $\delta^* = \delta_k$, and if $0 \leq c \leq c_{\lfloor \frac{B_1+3}{2} \rfloor}$, then $\delta^* = \delta_{\lfloor \frac{B_1+3}{2} \rfloor}$. Thus, the optimal server assignment policy is of double-threshold type.

Proof. We know that an optimal Markovian stationary deterministic policy exists by Theorem 9.1.8 of Puterman [56] since the state and action spaces are finite. Moreover, since $\mu_1 = 0$ implies that all service rates are zero, we can assume that $\mu_1 > 0$, so that the policies defined by the decision rules δ_k for $k \in \{1, \ldots, \lfloor \frac{B_1+3}{2} \rfloor\}$ result in Markov chains with a single recurrent class. If $\mu_1 > 0, \mu_2 > 0$ ($\mu_1 > 0, \mu_2 = 0$), the states $\{(0, 1, 2), \ldots, (B_1 + 3 - k, 1, 2), (k - 1, 2, 1), \ldots, (B_1 + 2, 2, 1)\}$ ($\{(k, 1, 2), \ldots, (B_1 + 3 - k, 1, 2), (k - 1, 2, 1), \ldots, (B_1 + 2 - k, 2, 1)\}$) form a recurrent class and the remaining states are transient. Note that the decision rule δ_0 as it is defined in Theorem 3, yields two recurrent classes which corresponds to a dedicated server assignment policy. Furthermore, due to homogeneity of the tasks, both recurrent classes yield the same long-run average throughput and result in no setups.

Consider the following LP:

$$\begin{aligned} \max & \sum_{x \in S} \sum_{a \in A_x} r(x, a) w(x, a) \\ s.t. & \sum_{a \in A_{x'}} w(x', a) - \sum_{x \in S} \sum_{a \in A_x} p(x'|x, a) w(x, a) = 0, \text{for all } x' \in S, \qquad (13) \\ & \sum_{x \in S} \sum_{a \in A_x} w(x, a) = 1, \\ & w(x, a) \geq 0, \text{for all } x \in S, a \in A_x, \end{aligned}$$

where, for all $x \in S$ and $a \in A_x$, $r(x,a)$ is the immediate reward of choosing action a in state x and $p(x'|x,a)$ is the transition probability from state x to x' if action a is chosen in state x. Since policies δ_k, $k \in \{1,\ldots,\lfloor\frac{B_1+3}{2}\rfloor\}$ result in a single recurrent class, every basic feasible solution that corresponds to a policy defined by one of the decision rules δ_k has at most one action $a_x \in A_x$ such that $w(x,a_x) > 0$ for each $x \in S$.

Furthermore, the optimal decision rule that corresponds to optimal dual solution w^* is

$$\delta^{w^*}(x) = \begin{cases} a & \text{if } w^*(x,a) > 0 \text{ for } x \in S_{w^*}, \\ a' & \text{for some } a' \text{ such that there exists a state } x' \in S_{w^*} \text{ for which } x' \\ & \text{is reachable from } x \text{ under action } a' \text{ for } x \in S \backslash S_{w^*}, \end{cases}$$

where $S_{w^*} = \{x \in S : \sum_{a \in A_x} w^*(x,a) > 0\}$. Note that an action a' that moves the process towards a recurrent state always exists. In particular, to move towards a recurrent state $(y,1,2)$ $((y,2,1))$ from a transient state (y',z'), we choose $a_{(1,1)(2,1)}$ $(a_{(2,1)(1,1)})$ if $y' < y$ $(y' > y)$, and choose $a_{(2,1)(1,1)}$ $(a_{(1,1)(2,1)})$ if $y' \geq y$ $(y' \leq y)$. As the following decision rule δ yields one of the recurrent classes yielded by δ_0 (i.e., $\{(0,1,2),\ldots,(B_1+2,1,2)\}$) and the actions in the remaining transient states are chosen so that the process is carried into the recurrent class, both decision rules result in the same long-run average profit and proving the optimality of the decision rule δ_0 is equivalent to proving the optimality of δ:

$$\delta(x) = \begin{cases} a_{(1,1)(2,1)} & \text{if } x = (s,1,2), s \leq B_1 + 2, \\ a_{(1,1)(2,1)} & \text{if } x = (0,2,1), \\ a_{(2,1)(1,1)} & \text{if } x = (s,2,1), 1 \leq s \leq B_1 + 2. \end{cases}$$

Since the decision rule δ also results in a single recurrent class, the corresponding basic feasible solution has at most one action $a_x \in A_x$ such that $w(x,a_x) > 0$ for each $x \in S$.

The Markov decision process that corresponds to our formulation is weakly communicating, since the deterministic stationary policy $(\delta_1)^\infty$ yields a closed recurrent class with no transient states if $\mu_2 > 0$, and it yields a single recurrent class with two transient states $(0,1,2), (B_1+2,2,1)$ that are transient under every policy if $\mu_2 = 0$. Therefore, we use the Linear Programming (LP) approach for weakly communicating Markov decision processes in order to show that the decision rule δ^* that is defined in Theorem 3 is optimal.

We prove the optimality of decision rule δ defined above, when setup cost $c \geq c_1$. Let w denote the basic solution of the LP (13) that corresponds to the policy $(\delta)^\infty$. Then the associated basis is

$$\begin{aligned} D = \{&w((0,1,2), a_{(1,1)(2,1)}), w((0,2,1), a_{(1,1)(2,1)}), w((1,1,2), a_{(1,1)(2,1)}), \\ &w((1,2,1), a_{(2,1)(1,1)}), w((2,1,2), a_{(1,1)(2,1)}), w((2,2,1), a_{(2,1)(1,1)}), \\ &\ldots, w((B_1+2,1,2), a_{(1,1)(2,1)}), w((B_1+2,2,1), a_{(2,1)(1,1)})\}. \end{aligned}$$

We let $\mathbf{B}$ be the coefficients for the elements of basis D in the constraint matrix and let $c_{\mathbf{B}}$ be the vector of coefficients for the basic variables in the objective function. Then, we have

$$c_{\mathbf{B}} = \{0, -2c\mu_1, \mu_2, \mu_1, \mu_2, \mu_1, \ldots, \mu_2, \mu_1\},$$

$$\mathbf{B} = \begin{pmatrix} \mu_1/q & 0 & -\mu_2/q & 0 & \ldots & \ldots & 0 & 0 & 0 & 0 \\ 0 & \mu_1/q & 0 & -\mu_1/q & \ddots & \ldots & \vdots & \vdots & \vdots & \vdots \\ -\mu_1/q & -\mu_1/q & (\mu_1+\mu_2)/q & 0 & \ddots & 0 & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & (\mu_1+\mu_2)/q & \ddots & -\mu_1/q & 0 & \vdots & \vdots & \vdots \\ \vdots & \vdots & -\mu_1/q & 0 & \ddots & 0 & -\mu_2/q & 0 & \vdots & \vdots \\ \vdots & \vdots & 0 & -\mu_2/q & \ddots & \mu_{11} & 0 & -\mu_1/q & 0 & \vdots \\ \vdots & \vdots & \vdots & 0 & \ddots & 0 & (\mu_1+\mu_2)/q & 0 & -\mu_2/q & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & -\mu_2/q & 0 & (\mu_1+\mu_2)/q & 0 & -\mu_1/q \\ 0 & 0 & 0 & 0 & \ldots & 0 & -\mu_1/q & 0 & \mu_2/q & 0 \\ 1 & 1 & 1 & 1 & \ldots & \ldots & 1 & 1 & 1 & 1 \end{pmatrix}$$

where q is the uniformization constant. Note that in the LP formulation (13), one of the constraints in the first set of constraints is redundant, so we eliminate the constraint that corresponds to the state $(B_1+2,2,1)$. To prove optimality, we only need to show that

$$\bar{c}_w = c_w - c_{\mathbf{B}}\mathbf{B}^{-1}v_w \leq 0 \tag{14}$$

for each nonbasic variable w, where v_w denotes the column corresponding to variable w in the constraint matrix and c_w is the coefficient of the variable w in the objective function (Theorem 3.1 of Bertsimas and Tsitsiklis [21]). Note that the set of nonbasic variables is

$$\begin{aligned} D' =& \{w((s,1,2),a)\colon a \in \{a_{(2,1)(1,1)}, a_{(1,1)(2,0)}, a_{(2,0)(1,1)}, a_{(2,1)(1,0)}, a_{(1,0)(2,1)}\}, 0 \leq s \leq B_1+2 \\ & w((s,2,1),a)\colon a \in \{a_{(1,1)(2,1)}, a_{(1,1)(2,0)}, a_{(2,0)(1,1)}, a_{(2,1)(1,0)}, a_{(1,0)(2,1)}\}, 1 \leq s \leq B_1+2 \\ & w((0,2,1),a)\colon a \in \{a_{(2,1)(1,1)}, a_{(1,1)(2,0)}, a_{(2,0)(1,1)}, a_{(2,1)(1,0)}, a_{(1,0)(2,1)}\}\}. \end{aligned}$$

In addition to the quantities $\bar{c}_w$, let us also define

$$\begin{aligned} \Delta_1(s,a) &= \bar{c}_{w((s,1,2),a)} - \bar{c}_{w((s+1,1,2),a)}, \\ \Delta_2(s,a) &= \bar{c}_{w((s,2,1),a)} - \bar{c}_{w((s+1,2,1),a)}, \\ \Delta_3(s,a) &= \bar{c}_{w((s,1,2),a)} - \bar{c}_{w((s,2,1),a)}. \end{aligned}$$

for every $s \in \{0,\ldots,B_1+2\}$ and action $a \in A$.

For states $(0,1,2)$, $(0,2,1)$ and $(B_1+2,2,1)$,

$$\begin{aligned} &\bar{c}_{w((0,1,2),a_{(2,1)(1,1)})} = -4c\mu_2 \leq 0, \\ &\bar{c}_{w((0,2,1),a_{(2,1)(1,1)})} = 0, \\ &\bar{c}_{w((B_1+2,2,1),a_{(1,1)(2,1)})} = \frac{-\mu_2 \sum_{i=0}^{B_1+1}\sum_{j=0}^{i}\mu_1^j\mu_2^{i-j}(\mu_1^{B_1+2-i}-\mu_2^{B_1+2-i})}{\sum_{i=0}^{B_1+2}\mu_i\mu_2^{B_1+2-i}} \leq 0. \end{aligned}$$

and the inequality (14) is satisfied for these non-basic variables for any $c \geq 0$. We also have

$$\Delta_2(s,a_{(1,1)(2,1)}) = \frac{(\mu_1+\mu_2)(\mu_1^{B_1+2-s}-\mu_2^{B_1+2-s})\sum_{i=0}^{s}\mu_1^i\mu_2^{s-i}}{\sum_{i=0}^{B_1+2}\mu_1^i\mu_2^{B_1+2-i}} \geq 0$$

for $s \in \{1, \ldots, B_1\}$, and

$$\bar{c}_{w((1,2,1),a_{(1,1)(2,1)})} = -\frac{(\mu_1+\mu_2)(\mu_1^{B_1+2}-\mu_2^{B_1+2})}{\sum_{i=0}^{B_1+2}\mu_1^i\mu_2^{B_1+2-i}} \leq 0.$$

Hence, the inequality (14) holds for all states $\{(1,2,1),(2,2,1),\ldots,(B_1+1,2,1)\}$ and action $a_{(1,1)(2,1)}$, for all values of c. Similarly, we have

$$\Delta_1(s, a_{(2,1)(1,1)}) = -\frac{(\mu_1+\mu_2)(\mu_1^{B_1+2-s}-\mu_2^{B_1+2-s})\sum_{i=0}^{s}\mu_1^i\mu_2^{s-i}}{\sum_{i=0}^{B_1+2}\mu_1^i\mu_2^{B_1+2-i}} \leq 0$$

for $s \in \{1, \ldots, B_1\}$, and

$$\begin{aligned}\bar{c}_{w((B_1+1,1,2),a_{(2,1)(1,1)})} =& -4c(\mu_1+\mu_2) + \frac{(\mu_1+\mu_2)}{(\mu_1-\mu_2)(\mu_1^{B_1+3}-\mu_2^{B_1+3})}\Bigg((B_1+1)(\mu_1^{B_1+4}-\mu_2^{B_1+4}) \\ &+ (B_1+2)(\mu_1\mu_2^{B_1+3}-\mu_1^{B_1+3}\mu_2) + \mu_1^2\mu_2^{B_1+2} - \mu_1^{B_1+2}\mu_2^2\Bigg).\end{aligned}$$

The inequality (14) holds for $\bar{c}_{w((B_1+1,1,2),a_{(2,1)(1,1)})}$, and thus for all $\bar{c}_{w((k,1,2),a_{(2,1)(1,1)})}$, $k \in \{1, \ldots, B_1+1\}$, when

$$c \geq \frac{\sum_{i=0}^{B_1}\sum_{j=0}^{i+1}\mu_1^j\mu_2^{i+1-j}(\mu_1^{B_1+1-i}-\mu_2^{B_1+1-i})}{4\sum_{i=0}^{B_1+2}\mu_2^i\mu_1^{B_1+2-i}}. \tag{15}$$

Furthermore,

$$\bar{c}_{w((B_1+2,1,2),a_{(2,1)(1,1)})} = \mu_1\Bigg(-4c - B_1 - 2 + \frac{2\mu_1\big((B_1+2)\mu_1^{B_1+3} - (B_1+3)\mu_1^{B_1+2}\mu_2 + \mu_2^{B_1+3}\big)}{(\mu_1-\mu_2)(\mu_1^{B_1+3}-\mu_2^{B_1+3})}\Bigg),$$

and $\bar{c}_{w((B_1+2,1,2),a_{(2,1)(1,1)})} \leq 0$ if

$$c \geq \frac{\sum_{i=0}^{B_1+1}\sum_{j=0}^{i}\mu_1^j\mu_2^{i-j}(\mu_1^{B_1+2-i}-\mu_2^{B_1+2-i})}{4\sum_{i=0}^{B_1+2}\mu_2^i\mu_1^{B_1+2-i}} = c_1. \tag{16}$$

Inequality (16) gives a tighter bound on c than inequality (15), hence the condition given in (14) holds for all non-basic variables corresponding to non-idling actions $a_{(1,1)(2,1)}$ and $a_{(2,1)(1,1)}$ given that $c \geq c_1$.

Note that we do not need to consider the idling actions $a_{(1,1)(2,0)}, a_{(2,0)(1,1)}$ in states $(B_1+2,1,2)$, $(B_1+2,2,1)$, and the idling actions $a_{(2,1)(1,0)}, a_{(1,0)(2,1)}$ in states $(0,1,2)$,

$(0,2,1)$, since they result in zero long-run average profit. For the idling actions $a_{(1,1)(2,0)}$ and $a_{(1,0)(2,1)}$, we have

$$\bar{c}_{w((0,1,2),a_{(1,1)(2,0)})} = 0,$$

$$\bar{c}_{w((0,2,1),a_{(1,1)(2,0)})} = 0,$$

$$\Delta_1(s, a_{(1,1)(2,0)}) = \frac{\mu_1^{B_1+2-s}\mu_2^{s+1}(\mu_1-\mu_2)}{\mu_1^{B_1+3}-\mu_2^{B_1+3}} \geq 0 \text{ for } s \in \{0,\ldots,B_1\},$$

$$\Delta_3(s, a_{(1,1)(2,0)}) = \frac{\sum_{i=1}^{s}(\mu_1^i-\mu_2^i)(\mu_1^{B_1+2-i}-\mu_2^{B_1+2-i})}{(\mu_1-\mu_2)(\mu_1^{B_1+3}-\mu_2^{B_1+3})} \geq 0 \text{ for } s \in \{1,\ldots,B_1+1\},$$

$$\bar{c}_{w((B_1+2,1,2),a_{(1,0)(2,1)})} = 0,$$

$$\Delta_1(s, a_{(1,0)(2,1)}) = -\frac{\mu_1^{B_1+2-s}\mu_2^{s+1}(\mu_1-\mu_2)}{\mu_1^{B_1+3}-\mu_2^{B_1+3}} \leq 0 \text{ for } s \in \{1,\ldots,B_1+1\},$$

$$\Delta_3(s, a_{(1,0)(2,1)}) = \frac{\sum_{i=1}^{s}(\mu_1^i-\mu_2^i)(\mu_1^{B_1+2-i}-\mu_2^{B_1+2-i})}{(\mu_1-\mu_2)(\mu_1^{B_1+3}-\mu_2^{B_1+3})} \geq 0 \text{ for } s \in \{1,\ldots,B_1+2\}.$$

Thus, the inequality (14) holds for all states and actions $a_{(1,1)(2,0)}$ and $a_{(1,0)(2,1)}$. For action $a_{(2,0)(1,1)}$, we have

$$\bar{c}_{w((0,1,2),a_{(2,0)(1,1)})} = -4c\mu_1,$$

$$\bar{c}_{w((0,2,1),a_{(2,0)(1,1)})} = 0,$$

$$\Delta_2(s, a_{(2,0)(1,1)}) = \frac{\mu_1^{B_1+2-s}\mu_2^{s+1}(\mu_1-\mu_2)}{\mu_1^{B_1+3}-\mu_2^{B_1+3}} \geq 0 \text{ for } s \in \{0,\ldots,B_1\},$$

and

$$\Delta_3(s, a_{(2,0)(1,1)}) = \Big(s(\mu_2^{B_1+4}+\mu_1^{B_1+3}\mu_2) + 4c(\mu_2^{B_1+4}-\mu_1^{B_1+3}\mu_2) - (\mu_1\mu_2^{B_1+4-s}+\mu_1^{B_1+3-s}\mu_2^2)\sum_{i=0}^{s-1}\mu_1^i\mu_2^{s-1-i}\Big) \Big/ \Big(\mu_1^{B_1+3}-\mu_2^{B_1+3}\Big)$$

for $s \in \{1,\ldots,B_1+1\}$. Similarly, for action $a_{(2,1)(1,0)}$,

$$\bar{c}_{w((B_1+2,2,1),a_{(2,1)(1,0)})} = 0,$$

$$\Delta_2(s, a_{(2,1)(1,0)}) = -\frac{\mu_1^{B_1+2-s}\mu_2^{s+1}(\mu_1-\mu_2)}{\mu_1^{B_1+3}-\mu_2^{B_1+3}} \leq 0 \text{ for } s \in \{1,\ldots,B_1+1\}$$

and

$$\Delta_3(s, a_{(2,1)(1,0)}) = \Big(s(\mu_2^{B_1+4} + \mu_1^{B_1+3}\mu_2) + 4c(\mu_2^{B_1+4} - \mu_1^{B_1+3}\mu_2) - (\mu_1\mu_2^{B_1+4-s} + \mu_1^{B_1+3-s}\mu_2^2) \sum_{i=0}^{s-1} \mu_1^i \mu_2^{s-1-i} \Big) \Big/ \Big(\mu_1^{B_1+3} - \mu_2^{B_1+3} \Big)$$

for $s \in \{1, \ldots, B_1 + 2\}$. Therefore $\Delta_3(s, a_{(2,0)(1,1)}) \leq 0$ for $s \in \{1, \ldots, B_1 + 1\}$ and $\Delta_3(s, a_{(2,1)(1,0)}) \leq 0$ for $s \in \{1, \ldots, B_1 + 2\}$ if

$$c \geq \frac{\sum_{i=0}^{s-1} \sum_{j=0}^{i} \mu_1^j \mu_2^{i-j} (\mu_1^{B_1+2-i} - \mu_2^{B_1+2-i})}{4\Big(\sum_{i=0}^{B_1+2} \mu_2^i \mu_1^{B_1+2-i} \Big)}, \text{ for } s \in \{1, \ldots, B_1 + 2\}. \tag{17}$$

Note that the inequality (16) is tighter than the set of inequalities (17). Thus, given $c \geq c1$, (14) holds for all non-basic variables, and the decision rule δ, or equivalently δ_0, is optimal when $c \geq c_1$. Note that not all non-basic variables have strictly negative reduced costs, in particular $\bar{c}_{w((0,2,1),a_{(2,1)(1,1)})} = 0$, indicating multiple optimal bases. In fact, any server assignment policy that ends up in one of the recurrent classes of policy δ_0 ($\{(0,1,2), \ldots, (B_1 + 2, 1, 2)\}$ or $\{(0,2,1), \ldots, (B_1 + 2, 2, 1)\}$ if $\mu_2 > 0$ and $(B_1 + 2, 1, 2)$ or $(0, 2, 1)$ if $\mu_2 = 0$) is also optimal.

For any optimal policy that yields the recurrent class $\{(0,1,2), \ldots, (B_1+2,1,2)\}$, action $a_{(1,1)(2,1)}$ should be chosen as the optimal action in states $\{(0,1,2), \ldots, (B_1 + 2, 1, 2)\}$ and in state $(s, 2, 1)$ for some $s \in \{0, \ldots, B_1 + 2\}$, and given that $\mu_2 > 0$, the following actions can be chosen in the remaining states:

- actions $\{a_{(1,1)(2,1)}, a_{(2,1)(1,1)}, a_{(1,1)(2,0)}, a_{(2,0)(1,1)}\}$ in state $(0, 2, 1)$,
- actions $\{a_{(1,1)(2,1)}, a_{(2,1)(1,1)}, a_{(1,0)(2,1)}, a_{(2,1)(1,0)}\}$ in state $(B_1 + 2, 2, 1)$,
- actions $\{a_{(1,1)(2,1)}, a_{(2,1)(1,1)}, a_{(1,0)(2,1)}, a_{(1,1)(2,0)}, a_{(2,0)(1,1)}\}$ in states $\{(1, 2, 1), \ldots, (s - 1, 2, 1)\}$,
- actions $\{a_{(1,1)(2,1)}, a_{(2,1)(1,1)}, a_{(1,0)(2,1)}, a_{(1,1)(2,0)}, a_{(2,1)(1,0)}\}$ in states $\{(s + 1, 2, 1), \ldots, (B_1 + 1, 2, 1)\}$.

(Note that if $\mu_2 = 0$, the actions $a_{(2,0)(1,1)}$ and $a_{(1,0)(2,1)}$ result in zero transition rates and cannot be optimal in any state, and the actions $a_{(1,1)(2,1)}$ and $a_{(2,1)(1,1)}$ cannot

be optimal in states $\{(s', 2, 1) \colon s' < s\}$ and $(B_1 + 2, 2, 1)$, respectively.) Similarly, for any optimal policy that yields the recurrent class $\{(0, 2, 1), \ldots, (B_1 + 2, 2, 1)\}$, action $a_{(2,1)(1,1)}$ should be chosen as the optimal action in states $\{(0, 2, 1), \ldots, (B_1 + 2, 2, 1)\}$ and state $(s, 1, 2)$ for some $s \in \{0, \ldots, B_1 + 2\}$, and given that $\mu_2 > 0$ the following actions can be chosen in the remaining states:

- actions $\{a_{(1,1)(2,1)}, a_{(2,1)(1,1)}, a_{(1,1)(2,0)}, a_{(2,0)(1,1)}\}$ in state $(0, 1, 2)$,
- actions $\{a_{(1,1)(2,1)}, a_{(2,1)(1,1)}, a_{(1,0)(2,1)}, a_{(2,1)(1,0)}\}$ in state $(B_1 + 2, 1, 2)$;
- actions $\{a_{(1,1)(2,1)}, a_{(2,1)(1,1)}, a_{(2,1)(1,0)}, a_{(2,0)(1,1)}, a_{(1,1)(2,0)}\}$ in states $\{(1, 1, 2), \ldots, (s - 1, 1, 2)\}$,
- actions $\{a_{(1,1)(2,1)}, a_{(2,1)(1,1)}, a_{(2,1)(1,0)}, a_{(2,0)(1,1)}, a_{(1,0)(2,1)}\}$ in states $\{(s + 1, 1, 2), \ldots, (B_1 + 1, 1, 2)\}$.

Optimality of the decision rule δ_k when $c_{k+1} \leq c \leq c_k$ is proved in a similar fashion. Once again, we let w denote the basic feasible solution that corresponds to policy $(\delta_k)^\infty$. Then the associated basis (D), coefficients of basic variables in the constraint matrix $(\mathbf{B})$, and coefficients of the basic variables in the objective function $(c_{\mathbf{B}})$ are as follows:

$$\begin{aligned} D = \{&w((0, 1, 2), a_{(1,1)(2,1)}), w((0, 2, 1), a_{(1,1)(2,1)}), \ldots, w((k - 1, 1, 2), a_{(1,1)(2,1)}), \\ &w((k - 1, 2, 1), a_{(1,1)(2,1)}), w((k, 1, 2), a_{(1,1)(2,1)}), w((k, 2, 1), a_{(2,1)(1,1)}), \ldots, \\ &w((B_1 + 2 - k, 1, 2), a_{(1,1)(2,1)}), w((B_1 + 2 - k, 2, 1), a_{(2,1)(1,1)}), \\ &w((B_1 + 1 - k, 1, 2), a_{(2,1)(1,1)}), w((B_1 + 1 - k, 2, 1), a_{(2,1)(1,1)}), \ldots, \\ &w((B_1 + 2, 1, 2), a_{(2,1)(1,1)}), w((B_1 + 2, 2, 1), a_{(2,1)(1,1)})\}, \end{aligned}$$

$$\mathbf{B}_{ij} = \begin{cases} \mu_1/q & \text{if } i=1, j=1, \\ \mu_1/q & \text{if } i=2, j=2, \\ -\mu_1/q & \text{if } i=3, j\in\{1,2\}, \\ -\mu_2/q & \text{if } i=j-2, j=2n-1 \text{ or } i=j-3, j=2n \text{ for } 1<n\le k, \\ -\mu_1/q & \text{if } i=j+2, j=2n-1 \text{ or } i=j+1, j=2n \text{ for } 1<n\le k, \\ (\mu_1+\mu_2)/q & \text{if } i=j, 2<i\le B_1 \\ -\mu_2/q & \text{if } i=j-2, j=2n+1 \text{ or } i=j+2, j=2n \text{ for } k<n\le B_1+3-k, \\ -\mu_1/q & \text{if } i=j+2, j=2n+1 \text{ or } i=j-2, j=2n \text{ for } k<n\le B_1+3-k, \\ -\mu_1/q & \text{if } i=j-1, j=2n+1 \text{ or } i=j-2, j=2n \text{ for } B_1+3-k<n\le B_1+1, \\ -\mu_2/q & \text{if } i=j+3, j=2n+1 \text{ or } i=j+2, j=2n \text{ for } B_1+3-k<n\le B_1+1, \\ -\mu_1/q & \text{if } i=2(B_1+3)-4, j\in\{2(B_1+3)-3, 2(B_1+3)-2\}, \\ -\mu_1/q & \text{if } i=2(B_1+3)-2, j\in\{2(B_1+3)-1, 2(B_1+3)\}, \\ \mu_1/q & \text{if } i=2(B_1+3)-1, j=2(B_1+3)-1, \\ 1/q & \text{if } i=2(B_1+3), 1\le j\le 2(B_1+3), \\ 0 & \text{otherwise,} \end{cases}$$

and

$$c_{\mathbf{B}}(x) = \begin{cases} 0 & \text{if } x=(0,1,2) \\ -2c\mu_1 & \text{if } x=(0,2,1) \\ \mu_2 & \text{if } x=(s,1,2), 0<s\le B_1+2-k, \\ \mu_1 & \text{if } x=(s,2,1), k\le s<B_1+2, \\ \mu_2-2c(\mu_1+\mu_2) & \text{if } x=(s,2,1), 0<s\le k, \\ \mu_1-2c(\mu_1+\mu_2) & \text{if } x=(s,2,1), B_1+2-k<s<B_1+2, \\ \mu_1-2c\mu_1 & \text{if } x=(B_1+2,1,2) \\ \mu_1 & \text{if } x=(B_1+2,2,1). \end{cases}$$

We show that the inequality (14) holds for all nonbasic variables when $c_{k+1} < c \le c_k$.

For the non-idling actions $a_{(1,1)(2,1)}$ and $a_{(2,1)(1,1)}$, we have

$$\bar{c}_{w((s,2,1),a_{(2,1)(1,1)})} = \bar{c}_{w((B_1+2-s,1,2),a_{(1,1)(2,1)})},$$

$$\bar{c}_{w((s,1,2),a_{(2,1)(1,1)})} = \bar{c}_{w((B_1+2-s,2,1),a_{(1,1)(2,1)})} \text{ where } s \in \{0,\dots,k-1\},$$

$$\bar{c}_{w((B_1+2-s,1,2),a_{(2,1)(1,1)})} = \bar{c}_{w((s,2,1),a_{(1,1)(2,1)})} \text{ where } s \in \{k,\dots,B_1+2\}.$$

Hence, it is sufficient to show that the inequality (14) holds for $\bar{c}_{w((s,2,1),a_{(2,1)(1,1)})}$, $\bar{c}_{w((s,1,2),a_{(2,1)(1,1)})}$ for $s \in \{0,\dots,k-1\}$ and $\bar{c}_{w((s,2,1),a_{(1,1)(2,1)})}$ where $s \in \{k,\dots,B_1+2\}$.

First, we look at states $\{(0,1,2),(0,2,1),\dots,(k-1,1,2),(k-1,2,1)\}$. We have

$$\Delta_3(0,a_{(2,1)(1,1)}) = -4c\mu_2,$$

$$\Delta_3(s,a_{(2,1)(1,1)}) = -4c(\mu_1+\mu_2)$$

for $0 < s \le k-1$. Moreover,

$$\Delta_2(s,a_{(2,1)(1,1)}) = \frac{(4+B_1+4c-2k)\mu_1^{B_1+1-s}(\mu_2-\mu_1)^3\mu_2^s(\mu_1+\mu_2)}{2(\mu_1-\mu_2)\big((4+B_1-2k)\mu_1^{B_1+3}-\mu_1^k\mu_2^k\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i}\big)} \le 0$$

for all $0 \le s \le k-2$. Hence $\bar{c}_{w((k-2,2,1),a_{(2,1)(1,1)})} \le 0$ implies that the inequality (14) holds for all $\bar{c}_{w((s,2,1),a_{(2,1)(1,1)})}, s \in \{0,\dots,k-2\}$. We have $\bar{c}_{w((k-2,2,1),a_{(2,1)(1,1)})} = \frac{\Psi_2}{\Psi_1}$, where

$$\Psi_1 = 2\big((4+B_1-2k)\mu_1^{B_1+3}-\mu_1^k\mu_2^k\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i}\big),$$

$$\begin{aligned}\Psi_2 =& -2\mu_1^k\mu_2^{B_1+4-k}-(4+B_1+4c-2k)\mu_1^{B_1+6-k}\mu_2^{k-2}+4c\mu_1^{B_1+4}\\ &-8c\mu_1^{B_1+3}\mu_2+(6+B_1+4c-2k)\mu_1^{B_1+4-k}\mu_2^k.\end{aligned}$$

Therefore $\bar{c}_{w((k-2,2,1),a_{(2,1)(1,1)})} \le 0$ holds when

$$c \le \frac{\sum_{i=0}^{B_1+3-2k}\sum_{j=0}^{i}\mu_1^{j+B_1+3-k}\mu_2^{i-j+k-1}+\sum_{i=0}^{B_1+3-2k}\sum_{j=0}^{i+1}\mu_1^{j+B_1+3-k}\mu_2^{i-j+k-1}}{4\Big(\mu_1^{B_1+4-k}\sum_{i=0}^{k-2}\mu_2^i\mu_1^{k-2-i}+\mu_1^{k-1}\sum_{i=0}^{k-3}\mu_2^i\mu_1^{B_1+3-k-i}\Big)}. \quad (18)$$

Note that the right hand side of the above inequality is positive, yielding a valid bound on setup cost c.

Similarly for state $(k-1,2,1)$, we have $\bar{c}_{w((k-1,2,1),a_{(2,1)(1,1)})} = \frac{\Psi_3}{(\mu_1-\mu_2)\Psi_1}$, where

$$\begin{aligned}\Psi_3 =&(6+B_1+4c-2k)\mu_1^{k+1}\mu_2^{B+4-1} + (4+B_1+4c-2k)\mu_1^{k}\mu_2^{B_1+5-k} + 8c\mu_1^{B_1+5+k}\\ &- (4+B_1+4c-2k)\mu_1^{B_1+6-k}\mu_2^{2k} - 8c\mu_1^{B_1+4}\mu_2 + (6+B_1+4c-2k)\mu_1^{B_1+5-k}\mu_2^{k}.\end{aligned}$$

Therefore $\bar{c}_{w((k-1,2,1),a_{(2,1)(1,1)})} \leq 0$ holds when

$$c \leq \frac{\sum_{i=0}^{B_1+3-2k}\sum_{j=0}^{i}\mu_1^j\mu_2^{i-j}(\mu_1^{B_1+3-k-i}\mu_2^{k-1} - \mu_1^{k-1}\mu_2^{B_1+3-k-i})}{4\Big(\mu_1^{B_1+4-k}\sum_{i=0}^{k-2}\mu_2^i\mu_1^{k-2-i} + \mu_1^{k-1}\sum_{i=0}^{B_1+3-k}\mu_2^i\mu_1^{B_1+3-k-i}\Big)} = c_k. \tag{19}$$

Since the bound given by inequality (19) is tighter than the bound given by inequality (18), the optimality condition given in (14) holds for all states $\{(0,1,2),(0,2,1),\ldots,(k-1,1,2),(k-1,2,1)\}$ and non-idling actions when (19) is satisfied.

We have $\Delta_2(s,a_{(1,1)(2,1)}) = \frac{\Psi_4}{\Psi_1} \geq 0$ for $s \in \{k,\ldots,B_1+2\}$, where

$$\begin{aligned}\Psi_4 =&(\mu_1+\mu_2)\big(2\mu_1^k\mu_2^k\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i} + 8c\mu_1^{B_1+3}\\ &- (B_1+4+4c-2k)(\mu_1^{B_1+2-s}\mu_2^{s+1} + \mu_1^{s+1}\mu_2^{B_1+2-s})\big).\end{aligned}$$

Hence, $\bar{c}_{w((k,2,1),a_{(1,1)(2,1)})} \leq 0$ implies that (14) holds for all $\bar{c}_{w((s,2,1),a_{(1,1)(2,1)})}$, $s \in \{k,\ldots,B_1+2\}$. We have $\bar{c}_{w((k,2,1),a_{(1,1)(2,1)})} = \frac{\Psi_5}{(\mu_1-\mu_2))\Psi_1}$, where

$$\begin{aligned}\Psi_5 =& -(\mu_1+\mu_2)\big((B_1+4+4c-2k)(\mu_1^{B_1+3-k}\mu_2^{k+1} - \mu_1^{k+1}\mu_2^{B_1+3-k})\\ &+ (B_1+2+4c-2k)(\mu_1^k\mu_2^{B_1+4-k} - \mu_1^{B_1+4-k}\mu_2^k) + 8c(\mu_1^{B_1+4} - \mu_1^{B_1+3}\mu_2)\big),\end{aligned}$$

and $\bar{c}_{w((k,2,1),a_{(1,1)(2,1)})} \leq 0$ holds when

$$c \geq \frac{\sum_{i=0}^{B_1+1-2k}\sum_{j=0}^{i}\mu_1^j\mu_2^{i-j}(\mu_1^{B_1+2-k-i}\mu_2^{k} - \mu_1^{k}\mu_2^{B_1+2-k-i})}{4\Big(\mu_1^{B_1+3-k}\sum_{i=0}^{k-1}\mu_2^i\mu_1^{k-1-i} + \mu_1^{k}\sum_{i=0}^{B_1+2-k}\mu_2^i\mu_1^{B_1+2-k-i}\Big)} = c_{k+1}. \tag{20}$$

Thus the inequality (14) holds for all states $\{(k,1,2),(k,2,1),\ldots,(B_1+2,1,2),(B_1+2,2,1)\}$ when (20) is satisfied.

For the idling actions $a_{(1,1)(2,0)}, a_{(2,0)(1,1)}, a_{(2,1)(1,0)}$, and $a_{(1,0)(2,1)}$, we have

$$\bar{c}_w((s,2,1),a_{(1,1)(2,0)}) = \bar{c}_w((B_1+2-s,1,2),a_{(2,1)(1,0)}),$$

$$\bar{c}_w((s,2,1),a_{(2,0)(1,1)}) = \bar{c}_w((B_1+2-s,1,2),a_{(1,0)(2,1)}),$$

for $s \in \{0, \ldots, B_1 + 2\}$. Moreover, actions $a_{(1,1)(2,0)}$ and $a_{(2,0)(1,1)}$ result in zero long-run average profit if chosen in states $(B_1 + 2, 1, 2)$ and $(B_1 + 2, 2, 1)$. Hence, it is enough to show the inequality (14) holds for $\bar{c}_w((s, 1, 2), a)$ and $\bar{c}_w((s, 2, 1), a)$ for $s \in \{0, \ldots, B_1 + 1\}$, $a \in \{a_{(1,1)(2,0)}, a_{(2,0)(1,1)}\}$.

For action $a_{(1,1)(2,0)}$ and states $(s, 1, 2)$, we have

$$\Delta_1(s, a_{(1,1)(2,0)}) = \frac{(B_1 + 4 + 4c - 2k)(\mu_1 - \mu_2)\mu_1^{B_1+2+k-s}\mu_2^{k+s+1}}{2((B_1 + 4 - 2k)\mu_1^{B_1+k+3}\mu_2^k - \mu_1^{2k}\mu_2^{2k}\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i})} \geq 0$$

for $s \in \{0, \ldots, B_1+1-k\}$ and $\bar{c}_w((0, 1, 2), a_{(1,1)(2,0)}) = 0$, hence $\bar{c}_w((s, 1, 2), a_{(1,1)(2,0)}) \leq 0$ for all $s \in \{0, \ldots, B_1 + 2 - k\}$. Similarly,

$$\Delta_1(s, a_{(1,1)(2,0)}) = \frac{(B_1 + 4 + 4c - 2k)(\mu_1 - \mu_2)\mu_1^{k+s+2}\mu_2^{B_1+1+k-s}}{2((B_1 + 4 - 2k)\mu_1^{B_1+k+3}\mu_2^k - \mu_1^{2k}\mu_2^{2k}\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i})} \geq 0$$

for $s \in \{B_1 + 3 - k, \ldots, B_1 + 1\}$, and $\bar{c}_w((B_1 + 3 - k, 1, 2), a_{(1,1)(2,0)}) \leq 0$ implies that $\bar{c}_w((s, 1, 2), a_{(1,1)(2,0)}) \leq 0$ for all $s \in \{B_1 + 3 - k, \ldots, B_1 + 1\}$. We have $\bar{c}_w((B_1 + 3 - k, 1, 2), a_{(1,1)(2,0)}) = \frac{\Psi_6}{(\mu_1-\mu_2)\Psi_1}$, where

$$\begin{aligned}\Psi_6 =&(B_1 + 4 + 4c - 2k)(\mu_1^{B_1+2}\mu_2^3 + \mu_1^{B_1+4-k}\mu_2^{k+1} - \mu_1^{B_1+5-k}\mu_2^k - \mu_1^{B_1+3}\mu_2^2)\\ &+ 8c(\mu_1^{B_1+4}\mu_2 - \mu_1^{B_1+3}\mu_2^2) + 2(\mu_1^{B_1+4-k}\mu_2^{k+1} - \mu_1^k\mu_2^{B_1+5-k}).\end{aligned}$$

Therefore $\bar{c}_w((B_1 + 3 - k, 1, 2), a_{(1,1)(2,0)}) \leq 0$ holds when

$$c \leq \frac{(B_1 + 4 - 2k)(\mu_1^{B_1+2+k}\mu_2^{k+2} + \mu_1^{B_1+4}\mu_2^{2k}) - 2\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i}}{4(2\mu_1^{B_1+3+k}\mu_2^{k+1} - \mu_1^{B_1+2+k}\mu_2^{k+2} - \mu_1^{B_1+4}\mu_2^{2k})}. \tag{21}$$

Note that the bound given by inequality (19) is tighter than the bound given by inequality (21). Thus, given that (19) holds, $\bar{c}_w((s, 1, 2), a_{(1,1)(2,0)}) \leq 0$ for all $s \in \{(0, \ldots, B_1 + 2\}$.

For action $a_{(1,1)(2,0)}$ and states $(s, 2, 1)$, we have

$\bar{c}_w((s, 2, 1), a_{(1,1)(2,0)}) = \bar{c}_w((s, 1, 2), a_{(1,1)(2,0)}) \leq 0$ for $s \in \{0, \ldots, k - 1\}$

$\bar{c}_w((s, 2, 1), a_{(1,1)(2,0)}) = \bar{c}_w((s, 1, 2), a_{(1,1)(2,0)}) - 4c\mu_1 \leq 0$ for $s \in \{B_1 + 3 - k, \ldots, B_1 + 1\}$

and $\Delta_3(s, a_{(1,1)(2,0)}) = \frac{\Psi_7}{(\mu_1-\mu_2)\Psi_1}$, for $s \in \{k, \ldots, B_1 + 2 - k\}$, where

$$\begin{aligned}\Psi_7 =&(4 + B_1 + 4c - 2k)(\mu_1^{B_1+4-s}\mu_2^{s+1} - \mu_1^{s+2}\mu_2^{B_1+3-s}) + 8c(s + 1 - k)(\mu_1^{B_1+5} - \mu_1^{B_1+4}\mu_2)\\ &+ (2 + B_1 + 4c - 2s)(\mu_1^{k+1}\mu_2^{B_1+4-k} - \mu_1^{B_1+5-k}\mu_2^{k}).\end{aligned}$$

We have $\Psi_7 \geq 0$ for all non-negative values of c when $\lceil \frac{B_1+2}{2} \rceil \leq s \leq B_1 + 2 - k$. Also, $\Psi_7 \geq 0$ if $c \geq \frac{\sum_{l=k+1}^{s+1} \psi_1(s,l)}{\sum_{l=k+1}^{s+1} \psi_2(l)}$ when $k \leq s \leq \lceil \frac{B_1}{2} \rceil$, where

$$\psi_1(s,l) = \sum_{i=s+1-l}^{B_1+2-l-s} \sum_{j=0}^{i} \mu_1^j \mu_2^{i-j} (\mu_1^{B_1+3-l-i}\mu_2^{l-1} - \mu_1^{l-1}\mu_2^{B_1+3-l-i}),$$

$$\psi_2(l) = 4\Big(\mu_1^{B_1+4-l} \sum_{i=0}^{l-2} \mu_2^i \mu_1^{l-2-i} + \mu_1^{l-1} \sum_{i=0}^{B_1+3-l} \mu_2^i \mu_1^{B_1+3-l-i}\Big).$$

Note that $\frac{\psi_1(k,k+1)}{\psi_2(k+1)} = c_{k+1}$, and for $s \in \{k, \ldots, \lceil \frac{B_1}{2} \rceil\}$,

$$\begin{aligned}\frac{\sum_{l=k+1}^{s+1} \psi_1(s,l)}{\sum_{l=k+1}^{s+1} \psi_2(l)} &\leq \frac{\sum_{l=k+1}^{s+1} \sum_{i=0}^{B_1+3-2l} \sum_{j=0}^{i} \mu_1^j \mu_2^{i-j} (\mu_1^{B_1+3-l-i}\mu_2^{l-1} - \mu_1^{l-1}\mu_2^{B_1+3-l-i})}{\sum_{l=k+1}^{s+1} \psi_2(l)}\\ &\leq c_{k+1}\end{aligned}$$

since c_k is decreasing in k. Thus, $\Delta_3(s, a_{(1,1)(2,0)}) \geq 0$ for $s \in \{k, \ldots, B_1 + 2 - k\}$, and $\bar{c}_w((s, 2, 1), a_{(1,1)(2,0)}) \leq 0$ for all $s \in \{0, \ldots, B_1 + 1\}$.

For action $a_{(2,0)(1,1)}$ and states $(s, 2, 1)$, we have

$$\Delta_2(s, a_{(2,0)(1,1)}) = \frac{(B_1 + 4 + 4c - 2k)(\mu_1 - \mu_2)\mu_1^{B_1+k-s+1}\mu_2^{k+s+2}}{2((B_1 + 4 - 2k)\mu_1^{B_1+k+3}\mu_2^k - \mu_1^{2k}\mu_2^{2k} \sum_{i=0}^{B_1+3-2k} \mu_1^i \mu_2^{B_1+3-2k-i})} \geq 0$$

for $s \in \{0, \ldots, k-3\}$ and $\bar{c}_w((0, 2, 1), a_{(2,0)(1,1)}) \leq 0$ implies that $\bar{c}_w((s, 2, 1), a_{(2,0)(1,1)}) \leq 0$ for all $s \in \{0, \ldots, k - 2\}$. We have $\bar{c}_w((0, 2, 1), a_{(2,0)(1,1)}) = \frac{\Psi_8}{\Psi_1}$, where

$$\begin{aligned}\Psi_8 =&(B_1 + 4 - 2k)(\mu_1^{B_1+2+k}\mu_2^{k+2} - \mu_1^{B_1+4+k}\mu_2^{k}) + 2\mu_1^{B_1+4}\mu_2^{2k} - 2\mu_1^{2k}\mu_2^{B_1+4}\\ &+ 4c(\mu_1^{B_1+2+k}\mu_2^{k+2} + \mu_1^{B_1+4+k}\mu_2^{k} - 2\mu_1^{B_1+3+k}\mu_2^{k+1}).\end{aligned}$$

Therefore $\bar{c}_w((0, 2, 1), a_{(2,0)(1,1)}) \leq 0$ holds when

$$c \leq \frac{(B_1 + 4 - 2k)(\mu_1 + \mu_2)\mu_1^{B_1+2+k}\mu_2^k - 2\mu_1^{2k}\mu_2^{2k} \sum_{i=0}^{B_1+3-2k} \mu_1^i \mu_{B_1+3-2k-i}}{4(\mu_1 - \mu_2)\mu_1^{B_1+2+k}\mu_2^k}. \qquad (22)$$

Similarly,

$$\Delta_2(s, a_{(2,0)(1,1)}) = \frac{(B_1+4+4c-2k)(\mu_1-\mu_2)\mu_1^{k+s+1}\mu_2^{B_1+k-s+2}}{2((B_1+4-2k)\mu_1^{B_1+k+3}\mu_2^k - \mu_1^{2k}\mu_2^{2k}\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i})} \geq 0$$

for $s \in \{k-1, \ldots, B_1\}$, and $\bar{c}_w((k-1,21), a_{(2,0)(1,1)}) \leq 0$ implies that $\bar{c}_w((s,2,1), a_{(2,0)(1,1)}) \leq 0$ for all $s \in \{k-1, \ldots, B_1+1\}$. We have $\bar{c}_w((k-1,2,1), a_{(2,0)(1,1)}) = \frac{\Psi_9}{(\mu_1-\mu_2)\Psi_1}$, where

$$\begin{aligned}\Psi_9 =& (B_1+4-4c-2k)(\mu_1^{B_1+4+k}\mu_2^{k+1} - \mu_1^{B_1+5+k}\mu_2^k) \\ &+ (B_1+4+4c-2k)(\mu_1^{2k}\mu_2^{B_1+5} - \mu_1^{2k+1}\mu_2^{B_1+4}) + 2\mu_1^{B_1+5}\mu_2^{2k} - 2\mu_1^{2k+1}\mu_2^{B_1+4}.\end{aligned}$$

Therefore $\bar{c}_w((k-1,2,1), a_{(2,0)(1,1)}) \leq 0$ holds when

$$c \leq \frac{(B_1+4-2k)(\mu_1^{B_1+4+k}\mu_2^k + \mu_1^{2k}\mu_2^{B_1+4}) - 2\mu_1^{2k+1}\mu_2^{2k}\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i}}{4(\mu_1-\mu_2)\mu_1^{2k}\mu_2^k\sum_{i=0}^{B_1+3-k}\mu_1^i\mu_2^{B_1+3-k-i}}. \tag{23}$$

Note that the bound given by inequality (19) is tighter than the bounds given by inequalities (22) and (23). Thus, given that $c \leq c_k$, $\bar{c}_w((s,2,1), a_{(2,0)(1,1)}) \leq 0$ for all $s \in \{(0, \ldots, B_1+2\}$.

For action $a_{(2,0)(1,1)}$ and states $(s,1,2)$, we have

$$\bar{c}_w((s,1,2), a_{(2,0)(1,1)}) = \bar{c}_w((s,2,1), a_{(2,0)(1,1)}) \leq 0 \text{ for } s \in \{B_1+3-k, \ldots, B_1+1\},$$

$$\bar{c}_w((s,1,2), a_{(2,0)(1,1)}) = \bar{c}_w((s,2,1), a_{(2,0)(1,1)}) - 4c\mu_1 \leq 0 \text{ for } s \in \{0, \ldots, k-1\},$$

and $\Delta_3(s, a_{(2,0)(1,1)}) = \frac{\Psi_{10}}{(\mu_1-\mu_2)\Psi_1}$, for $s \in \{k, \ldots, B_1+2-k\}$, where

$$\begin{aligned}\Psi_{10} =& \mu_1^{-s}\mu_2^{1-s}(-(4+B_1+4c-2k)\mu_1^{1+k+2s}\mu_2^{3+B_1+k} + (4+B_1+4c-2k)\mu_1^{3+B_1+k}\mu_2^{1+k+2s} \\ &+ \mu_1^{2k+s}\mu_2^{4+B_1+s}(2+B_1-4c-2s) - \mu_1^{4+B_1+s}\mu_2^{2k+s}(2+B_1-4c-2s) \\ &- 8c\mu_1^{4+B_1+k+s}\mu_2^{k+s}(3+B_1-k-s) + 8c\mu_1^{3+B_1+k+s}\mu_2^{1+k+s}(3+B_1-k-s)).\end{aligned}$$

We have $\Psi_{10} \leq 0$ for all non-negative values of c when $k \leq s \leq \lceil \frac{B_1}{2} \rceil$. Also, $\Psi_{10} \leq 0$ if $c \geq \frac{\sum_{l=k+1}^{B_1+3-s}\psi_3(s,l)}{\sum_{l=k+1}^{B_1+3-s}\psi_2(l)}$ when $\lceil \frac{B_1+2}{2} \rceil \leq s \leq B_1+2-k$, where

$$\psi_3(s,l) = \sum_{i=B_1+3-s-l}^{s-l}\sum_{j=0}^{i}\mu_1^j\mu_2^{i-j}(\mu_1^{B_1+3-l-i}\mu_2^{l-1} - \mu_1^{l-1}\mu_2^{B_1+3-l-i}).$$

Note that $\frac{\psi_3(B_1+2-k,k+1)}{\psi_1(k+1)} = c_{k+1}$, and for $s \in \{\lceil \frac{B_1+2}{2} \rceil, \ldots, B_1+2-k\}$,

$$\frac{\sum_{l=k+1}^{B_1+3-s} \psi_3(s,l)}{\sum_{l=k+1}^{B_1+3-s} \psi_2(l)} \leq \frac{\sum_{l=k+1}^{B_1+3-s} \sum_{i=0}^{B_1+3-2l} \sum_{j=0}^{i} \mu_1^j \mu_2^{i-j} (\mu_1^{B_1+3-l-i} \mu_2^{l-1} - \mu_1^{l-1} \mu_2^{B_1+3-l-i})}{\sum_{l=k+1}^{B_1+3-s} \psi_2(l)}$$

$$\leq c_{k+1}$$

since c_k is decreasing in k. It follows that $\Delta_3(s, a_{(2,0)(1,1)}) \leq 0$, for $s \in \{k, \ldots, B_1 + 2 - k\}$, and $\bar{c}_w((s,1,2), a_{(2,0)(1,1)}) \leq 0$ for all $s \in \{0, \ldots, B_1+1\}$.

Thus, the condition given in inequality (14) is satisfied for the policy $(\delta_k)^\infty$ and δ_k is the optimal decision rule when $c_{k+1} \leq c \leq c_k$. Note that for $k \geq 1$ both inequality (19) and inequality (20) yield valid bounds on setup cost c for the decision rule δ_k to be optimal.

Our developments so far imply that $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ is the optimal decision rule when $c_{\lfloor \frac{B_1+5}{2} \rfloor} \leq c \leq c_{\lfloor \frac{B_1+3}{2} \rfloor}$. However, we have $c_{\lfloor \frac{B_1+5}{2} \rfloor} \leq 0$. Since we have $c \geq 0$, $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ must be optimal for $0 \leq c \leq c_{\lfloor \frac{B_1+3}{2} \rfloor}$, completing the proof. □

Remark 5. *Note that if $\mu_1 = \mu_2$, the bound given by inequality (15) becomes $c \geq c_1 = 0$. Therefore, any decision rule that corresponds to a dedicated server assignment policy is optimal for all values of setup cost $c \geq 0$ when the servers are identical. Furthermore, if $\mu_1 = \mu_2$, inequality (19) yields $c \leq 0$ and inequality (20) yields $c \geq 0$, implying that the decision rules δ_k for $k \in \{1 \ldots \lfloor \frac{B_1+3}{2} \rfloor\}$ are optimal only if the setup cost is zero.*

Note that the optimal policy is not unique if we have $c = c_k$ for some $k \in \{1, \ldots, \lfloor \frac{B_1+3}{2} \rfloor\}$. Also, an optimal policy $(\delta^*)^\infty = (\delta_k)^\infty$ can be modified in the transient states as long as the actions chosen carry the process into the recurrent class yielded by the decision rule δ_k. Our result is consistent with the optimal policy characterized in Chapter 3 for the non-collaborative system without setup costs. If there are no setup costs and the tasks are homogeneous, then it is shown in Chapter 3 that the optimal policy is threshold type and the optimal threshold is $s^* \in \{\lfloor \frac{B_1+3}{2} \rfloor, \lceil \frac{B_1+3}{2} \rceil\}$.

Note that this earlier result corresponds to the optimality of the decision rule $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ for $c = 0$ in our model.

When the servers are identical, the optimal server assignment policy is dedicated and cross-training has no value. Our next result shows that it is strictly suboptimal to use a dedicated policy when $c < c_1$. We also quantify the improvement in the long-run average profit due to the cross-training of the servers with the following proposition.

Proposition 8. *If server 1 is strictly faster than server 2 (i.e., $\mu_1 > \mu_2$), dedicated server assignment policies are strictly suboptimal when $c < c_1$, and the possible improvement in the long-run average profit due to cross training decreases as the setup cost c increases.*

Proof. Let $P_{(\delta_0)^\infty}$ denote the long-run average profit that is attained by a dedicated server assignment policy. Then we have

$$P_{(\delta_0)^\infty} = \frac{\mu_1\mu_2(\mu_1^{B_1+2} - \mu_2^{B_1+2})}{\mu_1^{B_1+3} - \mu_2^{B_1+3}}.$$

Theorem 3 states that the optimal long-run average profit in a system with setup cost c such that $c_{k+1} \leq c \leq c_k$ is attained by the decision rule δ_k. Let $P_{(\delta_k)^\infty}(c)$ denote the long-run average profit attained by the decision rule δ_k for setup cost c. We have

$$\begin{aligned} P_{(\delta_k)^\infty}(c) =& \mu_1 \Bigg(2\mu_1^{2k}\mu_2^{4+B_1} + (4 + B_1 - 4c - 2k)\mu_1^{4+B_1+k}\mu_2^k - 2\mu_1^{4+B_1}\mu_2^{2k} \\ &+ 8c\mu_1^{3+B_1+k}\mu_2^{1+k} - (4 + B_1 + 4c - 2k)\mu_1^{2+B_1+k}\mu_2^{2+k} \Bigg) \\ &\Bigg/ 2\Bigg(\mu_1^{2k}\mu_2^{4+B_1} + (4 + B_1 - 2k)\mu_1^{4+B_1+k}\mu_2^k - \mu_1^{4+B_1}\mu_2^{2k} \\ &- (4 + B_1 - 2k)\mu_1^{3+B_1+k}\mu_2^{1+k} \Bigg). \end{aligned}$$

Then the improvement in the long run average profit due to cross training is

$P_{(\delta_k)^\infty}(c) - P_{(\delta_0)^\infty}$. Note that $P_{(\delta_0)^\infty}$ does not depend on the setup cost c. Furthermore,

$$\frac{\partial P_{(\delta_k)^\infty}(c)}{\partial c} = -\frac{2(\mu_1-\mu_2)\mu_2^k \mu_1^{B_1+2+k}}{(B_1+4-2k)\mu_1^{B_1+3+k}\mu_2^k - \mu_1^{2k}\mu_2^{2k}\sum_{i=0}^{B_1+3-2k}\mu_1^i \mu_2^{B_1+3-2k-i}} < 0$$

since $\mu_1 > \mu_2$ and $\mu_1^{B_1+3+k}\mu_2^k > \mu_1^{2k+i}\mu_2^{B_1+3-i}$ for all $i \in \{0,1,\ldots,B_1+3-2k\}$. Hence, $P_{(\delta_k)^\infty(c)} - P_{(\delta_0)^\infty}$ decreases as the setup cost c increases.

Evaluating $P_{(\delta_k)^\infty}(c) - P_{(\delta_0)^\infty}$ for $c = c_k$ and $c = c_{k+1}$, we get

$$P_{(\delta_k)^\infty}(c_k) - P_{(\delta_0)^\infty} = \frac{\mu_1^{3+B_1}(\mu_1-\mu_2)\left(\mu_1^k\mu_2 - \mu_1\mu_2^k\right)\left(\mu_1^{4+B_1}\mu_2^k - \mu_1^k\mu_2^{4+B_1}\right)}{\left(\mu_1^{3+B_1}-\mu_2^{3+B_1}\right)\left(2\mu_1^{4+B_1+k}\mu_2^{1+k} - \mu_1^{2k}\mu_2^{5+B_1} - \mu_1^{5+B_1}\mu_2^{2k}\right)},$$

$$P_{(\delta_k)^\infty}(c_{k+1}) - P_{(\delta_0)^\infty} = \frac{\mu_1^{3+B_1}(\mu_1-\mu_2)\left(\mu_1^k - \mu_2^k\right)\left(\mu_1^{3+B_1}\mu_2^k - \mu_1^k\mu_2^{3+B_1}\right)}{\left(\mu_1^{3+B_1}-\mu_2^{3+B_1}\right)\left(2\mu_1^{3+B_1+k}\mu_2^k - \mu_1^{2k}\mu_2^{3+B_1} - \mu_1^{3+B_1}\mu_2^{2k}\right)}.$$

Note that if we have $c = c_1$, the dedicated assignment policies and the double threshold policy $(\delta_1)^\infty$ are optimal, and we have $P_{(\delta_1)^\infty}(c_1) - P_{(\delta_0)^\infty} = 0$. Since $P_{(\delta_1)^\infty}(c) - P_{(\delta_0)^\infty}$ is decreasing with setup cost c, it follows that we must have $P_{(\delta_1)^\infty}(c) - P_{(\delta_0)^\infty} > 0$ for $c < c_1$. □

If the setup costs are prohibitively high, the dedicated server assignment policies are optimal, even if $\mu_2 = 0$ and hence the long-run average profit attained by a dedicated server assignment policy is zero. More specifically, if $\mu_2 = 0$, then (19) yields $c_1 = \frac{B_1+2}{2}$, and $c_k = 0$ for all $k \in \{2,\ldots,\lfloor\frac{B_1+3}{2}\rfloor\}$. Therefore it is optimal to use a dedicated server assignment policy if $c \geq \frac{B_1+2}{2}$ and it is optimal to use the double threshold policy $(\delta_1)^\infty$ if $0 \leq c \leq \frac{B_1+2}{2}$. In this case, the double threshold policy $(\delta_1)^\infty$ yields

$$P_{(\delta_1)^\infty}(c) = \mu_1\left(\frac{B_1+2-4c}{2(B_1+2)}\right).$$

Note that if we have $c = 0$ and $\mu_2 = 0$, any of the policies $(\delta_k)^\infty$, $k \in \{1,\ldots,\lfloor\frac{B_1+3}{2}\rfloor\}$ is optimal and the long-run average profit is $\frac{\mu_1}{2}$.

We showed that the optimal policy changes with respect to the setup cost c. If the rates of the servers are relatively close to each other and the setup costs are high, the optimal policy tends to avoid setups. In particular, if $\mu_1 = \mu_2$, we have $c_k = 0$ for

all $k \in \{1, \ldots, \lfloor \frac{B+3}{2} \rfloor\}$, and it is optimal to use a dedicated server assignment policy to avoid setups completely, since a server reassignment cannot result in an increase in profits. The following result show that when $\mu_1 > \mu_2$, the optimal policy takes the setup costs into account and is different from the optimal policy for a system with no setup costs, unless $c \leq c_{\lfloor \frac{B_1+3}{2} \rfloor}$.

Proposition 9. *If server 1 is strictly faster than server 2 (i.e., $\mu_1 > \mu_2$), the improvement in the long-run average profit due to taking setup costs into account is increasing without a bound as the setup cost c increases, and the decision rule $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ is strictly suboptimal when $c > c_{\lfloor \frac{B_1+3}{2} \rfloor}$.*

Proof. It follows from Theorem 3 that the decision rule $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ is optimal for systems with no setup costs. Let $P_{(\delta_{\lfloor \frac{B_1+3}{2} \rfloor})^\infty}(c)$ denote the long-run average profit attained by the decision rule $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ in a system with setup cost c and let $P_{(\delta_k)^\infty}(c)$ be the optimal profit for the same system when $c_{k+1} \leq c \leq c_k$, as in the proof of Proposition 8. We have

$$\frac{\partial P_{(\delta_k)^\infty}(c)}{\partial c} - \frac{\partial P_{(\delta_{\lfloor \frac{B_1+3}{2} \rfloor})^\infty}(c)}{\partial c} = 2\mu_1^{B_1+2}(\mu_1 - \mu_2)\Big(\frac{1}{\Gamma_1} - \frac{1}{\Gamma_2}\Big),$$

where

$$\Gamma_1 = (B_1 + 4 - 2\lfloor \frac{B_1+3}{2} \rfloor)\mu_1^{B_1+3} - \mu_1^{\lfloor \frac{B_1+3}{2} \rfloor}\mu_2^{\lfloor \frac{B_1+3}{2} \rfloor} \sum_{i=0}^{B_1+3-2\lfloor \frac{B_1+3}{2} \rfloor} \mu_1^i \mu_2^{B_1+3-2\lfloor \frac{B_1+3}{2} \rfloor - i}$$

and

$$\Gamma_2 = (B_1 + 4 - 2k)\mu_1^{B_1+3} - \mu_1^k \mu_2^k \sum_{i=0}^{B_1+3-2k} \mu_1^i \mu_2^{B_1+3-2k-i}.$$

Furthermore,

$$\begin{aligned}\Gamma_1 - \Gamma_2 = \mu_1^k \mu_2^k \sum_{i=0}^{\lfloor \frac{B_1+3}{2} \rfloor - k - 1} \mu_1^i \mu_2^{B_1+3-2k-i} + \mu_1^k \mu_2^k \sum_{i=B_1+4-\lfloor \frac{B_1+3}{2} \rfloor - k}^{B_1+3-2k} \mu_1^i \mu_2^{B_1+3-2k-i} \\ - \mu_1^{B_1+3}(2\lfloor \frac{B_1+3}{2} \rfloor - 2k) \\ \leq 0,\end{aligned}$$

since $k \leq \lfloor \frac{B_1+3}{2} \rfloor$ and $\mu_1 > \mu_2$. Note that for $k = \lfloor \frac{B_1+3}{2} \rfloor$, the decision rule $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ is optimal and we have $\Gamma_1 - \Gamma_2 = 0$, $\frac{\partial P_{(\delta_k)^\infty}(c)}{\partial c} - \frac{\partial P_{(\delta_{\lfloor \frac{B_1+3}{2} \rfloor})^\infty}(c)}{\partial c} = 0$. For any $k < \lfloor \frac{B_1+3}{2} \rfloor$, we have $\Gamma_1 - \Gamma_2 < 0$ and the improvement in the long run average profit due to taking setup cost into account is strictly increasing as the setup cost c increases. In particular, if the setup cost is large enough so that a dedicated server assignment policy is optimal (i.e., $c > c_1$, $k = 1$), the optimal profit is constant with respect to c and the profit attained by $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ decreases as the setup cost c increases since the setup costs are incurred at recurrent states. Thus $P_{(\delta_k)^\infty}(c) - P_{(\delta_{\lfloor \frac{B_1+3}{2} \rfloor})^\infty}(c)$ increases without a bound.

To show the suboptimality of the decision rule $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ for $c > c_{\lfloor \frac{B_1+3}{2} \rfloor}$, we consider a system where $c_{\lfloor \frac{B_1+3}{2} \rfloor} \leq c \leq c_{\lfloor \frac{B_1+3}{2} \rfloor - 1}$ so that $k = \lfloor \frac{B_1+3}{2} \rfloor - 1$ and the decision rule $\delta_{\lfloor \frac{B_1+3}{2} \rfloor - 1}$ is optimal. Note that if $c = c_{\lfloor \frac{B_1+3}{2} \rfloor}$, $\delta_{\lfloor \frac{B_1+3}{2} \rfloor}$ is also optimal for this system and we must have $P_{(\delta_k)^\infty}(c) - P_{(\delta_{\lfloor \frac{B_1+3}{2} \rfloor})^\infty}(c) = 0$. On the other hand, $P_{(\delta_k)^\infty}(c) - P_{(\delta_{\lfloor \frac{B_1+3}{2} \rfloor})^\infty}(c)$ is strictly increasing with c since $k < \lfloor \frac{B_1+3}{2} \rfloor$. Thus, we have $P_{(\delta_k)^\infty}(c) - P_{(\delta_{\lfloor \frac{B_1+3}{2} \rfloor})^\infty}(c) > 0$ for $c > c_{\lfloor \frac{B_1+3}{2} \rfloor}$, completing the proof. □

So far we have characterized the optimal policy for systems with a finite buffer. The following result shows the behavior of the optimal policy as the buffer size grows larger.

Proposition 10. *As the buffer size B_1 increases, c_k tends to infinity, and furthermore*

$$\lim_{B_1 \to \infty} \frac{c_k}{B_1 + 4 - 2k} = \frac{\mu_2^{k-1}}{8\mu_1^{k-1} - 4\mu_2^{k-1}}.$$

Proof. We prove the result by directly computing the limit. We have

$$\lim_{B_1 \to \infty} \frac{c_k}{B_1 + 4 - 2k} = \lim_{B_1 \to \infty} \frac{\mu_1^{k-1}\mu_2^{B_1+4-k} + \mu_1^{B_1+4-k}\mu_2^{k-1} - \frac{2\mu_1^k\mu_2^k}{4+B_1-2k} \sum_{i=0}^{B_1+3-2k} \mu_1^i \mu_2^{B_1+3-2k-i}}{4(2\mu_1^{B_1+3} - \mu_{k-1}\mu_2^{B_1+4-k} - \mu_1^{B_1+4-k}\mu_2^{k-1})}.$$

To compute the above limit, we first compute the following

$$\lim_{B_1\to\infty} \frac{\mu_1^{k-1}\mu_2^{B_1+4-k} + \mu_1^{B_1+4-k}\mu_2^{k-1} - \frac{2\mu_1^k\mu_2^k}{4+B_1-2k}\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i}}{8\mu_1^{B_1+3}} \tag{24}$$

$$= \lim_{B_1\to\infty}\Big(\frac{\mu_2^{B_1+4-k}}{8\mu_1^{B_1+4-k}} + \frac{\mu_2^{k-1}}{8\mu_1^{k-1}} - \frac{\frac{2\mu_1^k\mu_2^k}{4+B_1-2k}\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i}}{8\mu_1^{B_1+3}}\Big)$$

$$= \frac{\mu_2^{k-1}}{8\mu_1^{k-1}} - \lim_{B_1\to\infty}\frac{\frac{2\mu_1^k\mu_2^k}{4+B_1-2k}\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i}}{8\mu_1^{B_1+3}}$$

$$= \frac{\mu_2^{k-1}}{8\mu_1^{k-1}} - (\lim_{B_1\to\infty}\frac{\mu_2^k}{4\mu_1^k(4+B_1-2k)})(\lim_{B_1\to\infty}\frac{\sum_{i=0}^{B_1+3-2k}\mu_1^i\mu_2^{B_1+3-2k-i}}{\mu_1^{B_1+3-2k}})$$

$$= \frac{\mu_2^{k-1}}{8\mu_1^{k-1}} - (0)\Big(\frac{1}{1-\frac{\mu_2}{\mu_1}}\Big)$$

$$= \frac{\mu_2^{k-1}}{8\mu_1^{k-1}}.$$

We also have,

$$\lim_{B_1\to\infty}\frac{8\mu_1^{B_1+3}}{4(2\mu_1^{B_1+3} - \mu_1^{k-1}\mu_2^{B_1+4-k} - \mu_1^{B_1+4-k}\mu_2^{k-1})} = \frac{2\mu_1^{k-1}}{2\mu_1^{k-1} - \mu_2^{k-1}}. \tag{25}$$

Using (24) and (25) we get

$$\lim_{B_1\to\infty}\frac{c_k}{B_1+4-2k} = \frac{\mu_2^{k-1}}{8\mu_1^{k-1} - 4\mu_2^{k-1}},$$

completing the proof. □

Proposition 10 implies that the setup cost values for which the decision rule δ_k is optimal becomes larger as the buffer size B_1 increases for any given value of k, as well as the cost interval $[c_{k+1}, c_k]$ since we have

$$\lim_{B_1\to\infty}\frac{c_k - c_{k+1}}{B_1+4-2k} = \frac{8\mu_1^k\mu_2^{k-1} - 8\mu_1^{k-1}\mu_2^k}{(8\mu_1^k - 4\mu_2^k)(8\mu_1^{k-1} - 4\mu_2^{k-1})} > 0.$$

Furthermore, for large enough buffer sizes, the boundary cost values c_k are proportional to $B_1 + 4 - 2k$. Note that number of values k can take for a decision rule δ_k increases as the buffer size B_1 increases and k can be as large as $\lfloor\frac{B_1+3}{2}\rfloor$ since $k \in \{1, \ldots, \lfloor\frac{B_1+3}{2}\rfloor\}$. Hence the result given in Proposition 10 is only meaningful for a fixed value of k. The following corollary shows that cross-training is beneficial for two-station systems with setup costs, as long as the buffer size is large enough.

Corollary 2. *For systems with two stations, homogeneous tasks, and any fixed setup cost c, the dedicated server assignment policies are strictly suboptimal if the buffer size B_1 is large enough.*

Proof. It follows from Proposition 8 that the dedicated decision rule δ_0 is strictly suboptimal when $c < c_1$. Furthermore, Proposition 10 states that c_k tends to infinity as the buffer size B_1 grows. Thus, there must exist a buffer size B' so that for any buffer size $B_1 \geq B'$, $c < c_1$ so that a dedicated server assignment policy is no longer optimal. □

4.5 Numerical Results

For systems with homogeneous tasks and constant setup costs, Theorem 3 characterizes the optimal server allocation policy. Furthermore, if each server is the fastest at a different task, Proposition 6 states that a dedicated server assignment policy is optimal. In this section, we develop heuristic server assignment policies for systems with non-homogeneous tasks and/or non-constant setup costs that use our results from Section 4.4. We evaluate the performance of these heuristics through a numerical study and present our results.

4.5.1 Systems with Non-Homogeneous Tasks and Constant Setup Costs

We consider a two-station system with non-homogeneous tasks and a dominating server. In order to develop our heuristics, we approximate this system by a homogeneous counterpart. In particular, we use two different approximations. The first approximation uses *homogenized rates* (HR) where we use the approximate rates $\mu_1 = \frac{\mu_{11}+\mu_{12}}{2}$, $\mu_2 = \frac{\mu_{21}+\mu_{22}}{2}$, and the second approximation uses *homogenized service times* (HST) where we approximate the rates as $\mu_1 = \frac{1}{\frac{1}{2}(\frac{1}{\mu_{11}}+\frac{1}{\mu_{12}})}$, $\mu_2 = \frac{1}{\frac{1}{2}(\frac{1}{\mu_{21}}+\frac{1}{\mu_{22}})}$. For each approximation method, we construct a corresponding *homogenized heuristic* policy (HR or HST) as follows. For the approximate system with service rates μ_1 and

μ_2, we compute the bounds $c_k, k \in \{1, \ldots, \lfloor \frac{B+3}{2} \rfloor\}$ as defined in (12). Using Theorem 3 and the bounds computed, we identify the server allocation policy that is optimal for the approximate system, and use it as a heuristic policy in the original system with non-homogeneous tasks.

We use the best dedicated policy as our benchmark policy and also include four *Pick-the-Best* (P-t-B) type of policies that pick the best performing policy among a set of policies for a given system. The first policy of this type picks the best performing policy among the two homogenized heuristics and dedicated policies, whereas the remaining three policies compare the best dedicated policy with the HR heuristic, the best dedicated policy with the HST heuristic, and the two homogenized heuristics with each other, respectively.

We evaluated performances of the policies described through a numerical study. For the experiments, 1000 sets of service rates were generated using a uniform distribution $U(1, 20)$ by discarding the sets without a dominating server. For each set of rates, buffer sizes $\{1, \ldots, 10\}$ were used. Note that in a two station non-collaborative system, if the cost of a single setup is c, a total setup cost of $2c$ is incurred every time servers switch places. Long-run average profit attained by the heuristic policies and the best dedicated policy is evaluated for setup cost values ranging from $2c = 1$ to $2c = 20$ with increments of 0.5. For each set of parameters, the long-run average throughput attained by the optimal server allocation policy was computed using the policy iteration algorithm. Figures 1 through 3 show the average performances of the optimal policy, best dedicated policy and two P-t-B policies (i.e., P-t-B (HR-HST) and P-t-B (all)) evaluated for setup cost values $2c = 1, 2c = 5$, and $2c = 20$ as a function of buffer size $B_1 \in \{1, \ldots, 10\}$. A comparison of the 95% confidence intervals for the long-run average profit attained by the optimal, best dedicated and all heuristic policies is given in Tables 25, 26, and 27 in Appendix A.

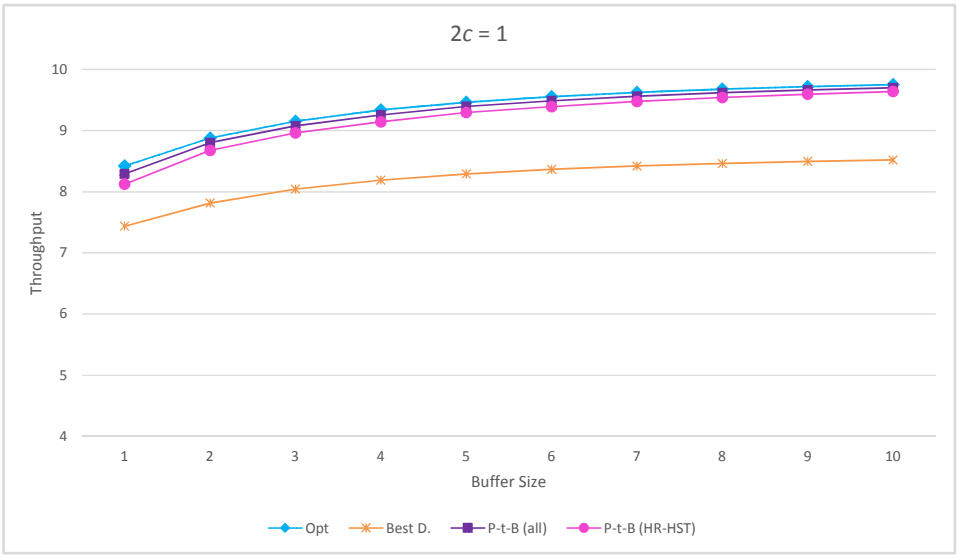

Figure 1: Non-Homogeneous Tasks with Constant Setup Cost ($2c = 1$)

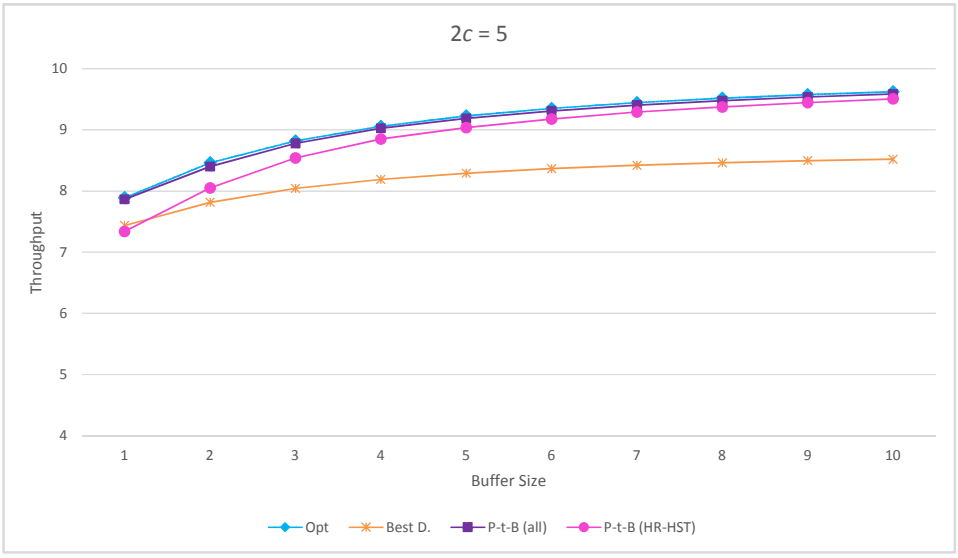

Figure 2: Non-Homogeneous Tasks with Constant Setup Cost ($2c = 5$)

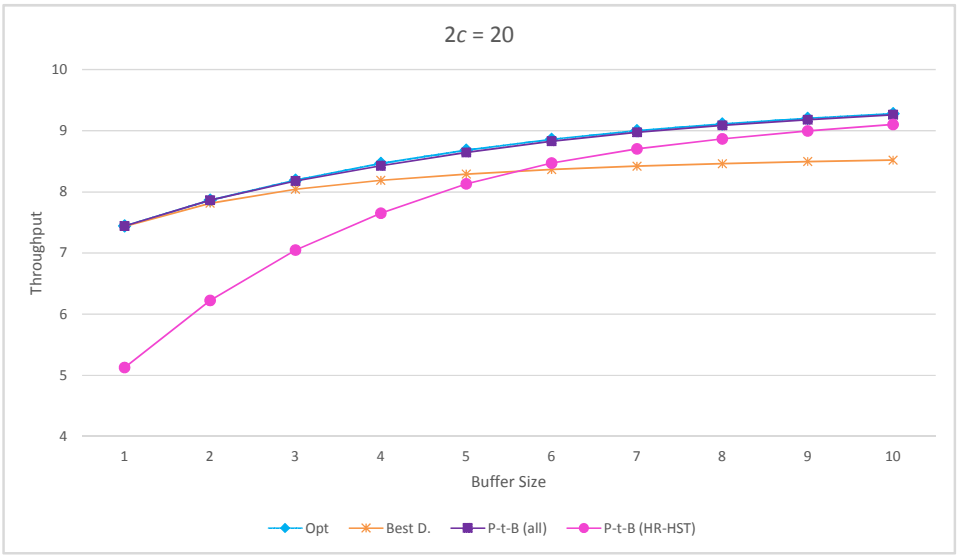

Figure 3: Non-Homogeneous Tasks with Constant Setup Cost ($2c = 20$)

The heuristic server assignment policies that use *homogenized rates* and *homogenized service times* perform almost identically. These policies yield near-optimal profits when the buffer size is large. In particular, they achieve at least 98% of the optimal profit when $B_1 = 10$ for setup cost values $2c = 1, 2c = 5, 2c = 20$. For smaller buffer sizes, the performance of the heuristics shows greater dependability on the magnitude of the setup costs and the optimality gap increases as the setup cost increases. For systems with $2c = 1$, the homogenized heuristics yield 96% of the optimal profit when $B_1 = 1$, whereas for $2c = 20$ and $B_1 = 1$ they achieve 67% of the optimal profit. Furthermore, the profits attainable by the best dedicated policy surpasses the profits attainable by the homogenized heuristics when setup costs are high and the buffer size is small, as suggested by our earlier result in Proposition 7. Table 27 shows that the best dedicated policy is actually near-optimal for small buffer sizes when the setup costs are relatively high.

These observations suggest that *Pick-the-Best* server assignment policies that

choose the best performing policy among the homogenized heuristics and the dedicated policies might perform near-optimal for large range of instances. Our results depicted in Figures 1, 2, and 3 show that while picking the best performing policy between the two homogenized heuristics does not lead to significant improvement, the (P-t-B) policy that compares both homogenized heuristics and the best dedicated policy is near-optimal. In fact, performances of all P-t-B policies that include the best dedicated policy in the comparison set are almost identical; therefore, picking the best performing policy between the best dedicated policy and one of the heuristic policies is sufficient. The average optimality gap for these policies is less than 1.6% for $c = 1$ and it is less than 0.32% for setup cost values $2c = 5, 2c = 10$ and $2c = 20$ at all buffer sizes. Hence, choosing the best policy between the homogenized and dedicated policies greatly narrows the optimality gap for all systems.

4.5.2 Systems with Non-Constant Setup Costs

In this section, we develop heuristic policies for systems with (non-constant) setup costs that depend on the positions of the servers before and after the server reassignment. In particular, a setup cost of $c(1,2) \geq 0$ is incurred if initially server 1 is at station 1, server 2 is at station 2 and the servers switch places. Similarly, a setup cost of $c(2,1) \geq 0$ is incurred if initially server 1 is at station 2 and server 2 is at station 1.

Our heuristics are composed of two approximations. If the tasks are non-homogeneous, we approximate the system by a homogeneous counterpart using the two methods described in Section 4.5.1 (i.e., *homogenized rates* and *homogenized service times*). We also approximate the server reassignment thresholds for the homogenized system using five different methods. The first four of our methods replace the non-constant setup costs with a constant value which is then compared with the values c_k where $k \in \{1, \ldots, \lfloor \frac{B+3}{2} \rfloor\}$. In particular, we use *average cost* (i.e., $2c = \frac{c(1,2)+c(2,1)}{2}$), *sum of the*

costs (i.e., $2c = c(1,2)+c(2,1)$), *maximum of the costs* (i.e., $2c = \max\{c(1,2), c(2,1)\}$) and *minimum of the costs* (i.e., $2c = \min\{c(1,2), c(2,1)\}$) to find a threshold k that satisfies $c_{k+1} \leq c \leq c_k$. The fifth method uses both setup cost values c_1 and c_2 to compute two threshold values k_1, k_2 such that $c_{k_1+1} \leq c_1 \leq c_{k_1}$ and $c_{k_2+1} \leq c_2 \leq c_{k_2}$. These threshold values k_1 and k_2 are used to characterize when the servers should switch stations given the current server assignment. In particular, if currently server 1 is at the first station and server 2 is at the second station, servers only switch places when the number of jobs in the buffer goes above $B_1 + 2 - k_1$. Similarly, if currently server 1 is at the second station and the server 2 is at the first station, server switch places when the number of jobs in the buffer goes below k_2. Using these approximations in conjunction with the ones described in Section 4.5.1, we developed 10 heuristic policies for systems with non-homogeneous tasks and setup costs, and 5 heuristic policies for systems with homogeneous tasks and setup costs.

Two sets of numerical experiments are presented in this section. The first set of experiments are run for systems with homogeneous tasks and non-constant setup costs, whereas the second set of experiments are run for systems with non-homogeneous tasks and non-constant setup costs. As in Section 4.5.1, when the tasks are non-homogeneous, the service rates are generated so that there is a dominating server since the optimal policy is dedicated otherwise. Since our results for systems with non-homogeneous tasks suggest that the performances of the two homogenized heuristics (i.e., HR and HST) are similar, in this section we only include the results for the heuristics that use *homogenized rates*. Performances of all heuristic policies including the ones that use *homogenizes service times* are reported in Appendix B.

For each set of experiments, 1000 sets of service rates were generated using a uniform distribution $U(1, 20)$. For each set of rates, buffer sizes $\{1, \ldots, 10\}$ were used. Long-run average profit attained by the heuristic policies and the best dedicated policy is evaluated for randomly generated setup cost values $c(1,2), c(2,1)$, using uniform

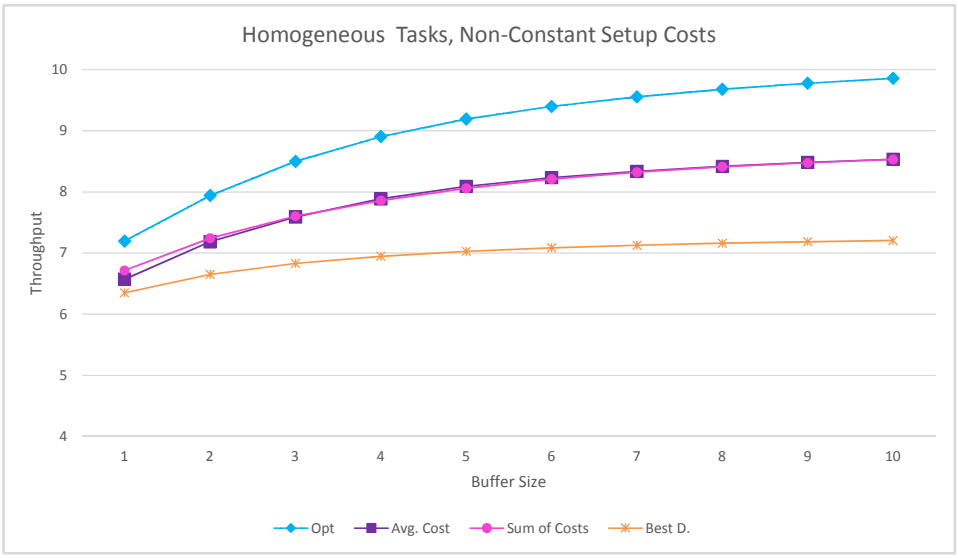

Figure 4: Homogeneous Tasks with Non-Constant Setup Costs

distribution $U(1, 20)$. For each set of parameters, the long-run average throughput attained by the optimal server allocation policy is computed using the policy iteration algorithm. Figures 4 and 5 depict the performances of the optimal policy, the best dedicated policy, and two heuristics that use *average cost* and *sum of the costs*. Tables 28 and 29 provide the 95% confidence intervals for the long-run average profit attained by the optimal policy, the best dedicated policy, and all heuristic policies.

For systems with homogeneous tasks and non-constant setup costs, all of the heuristics we developed attains 88% of the optimal long-run average profit when averaged over different buffer sizes. The best dedicated policy on average achieves 77% of the maximum attainable throughput. Although the differences between performances of the heuristic policies are small, if the buffer size is large, *average cost* heuristic performs the best with an optimality gap of 11% to 13%, whereas if the buffer size is small *sum of the costs* heuristic performs the best among the policies developed with an optimality gap of 7% to 10%. Note that the *sum of the costs* heuristic is more likely to yield a dedicated server assignment policy than the *average*

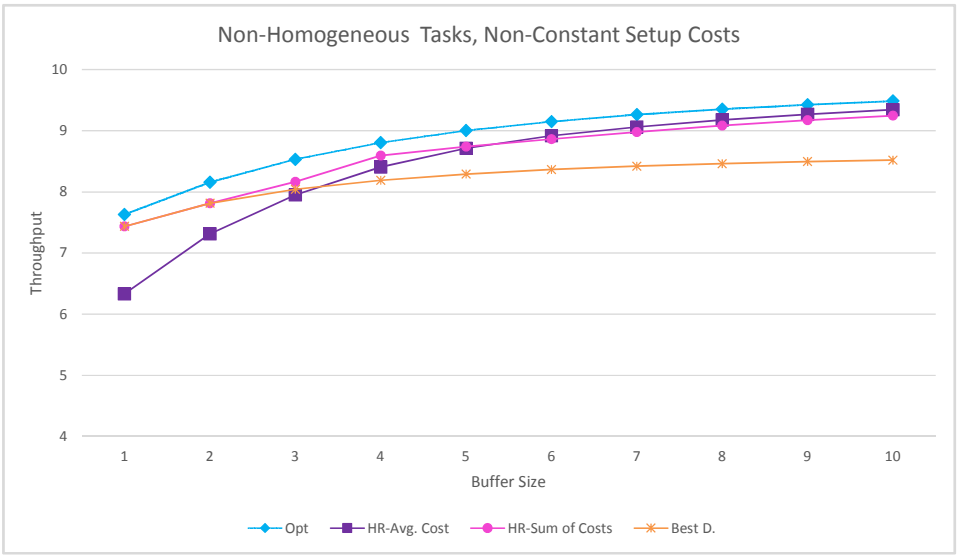

Figure 5: Non-Homogeneous Tasks with Non-Constant Setup Costs

cost heuristic, since it uses a larger setup cost value for a given instance of the system. These observations lead us to devise *pick-the-best* heuristics similar to the ones in Section 4.5.1. For each heuristic server assignment policy we developed earlier, we devise a new heuristic policy that picks the best performing policy between the given heuristic policy and the dedicated policies (i.e., each P-t-B heuristic compares one heuristic policy from Table 28 and the best dedicated policy). Figure 6 shows the performances of P-t-B policies for *average cost* and *sum of the costs*. When the tasks are homogeneous, P-t-B heuristics perform similar to each other at all buffer sizes, and they reduce the optimality gap attained by the original heuristics by 2% when the buffer size is small. Table 30 in Appendix B summarizes these results.

For systems with non-homogeneous tasks and non-constant setup costs, in contrast to systems with homogeneous tasks and non-constant setup costs, the performance of the best dedicated policy is improved and it surpasses all heuristic policies except *homogenized rates-sum of the costs* for small buffer sizes. The improvement in the performance of the best dedicated policy can be explained by the differences between the

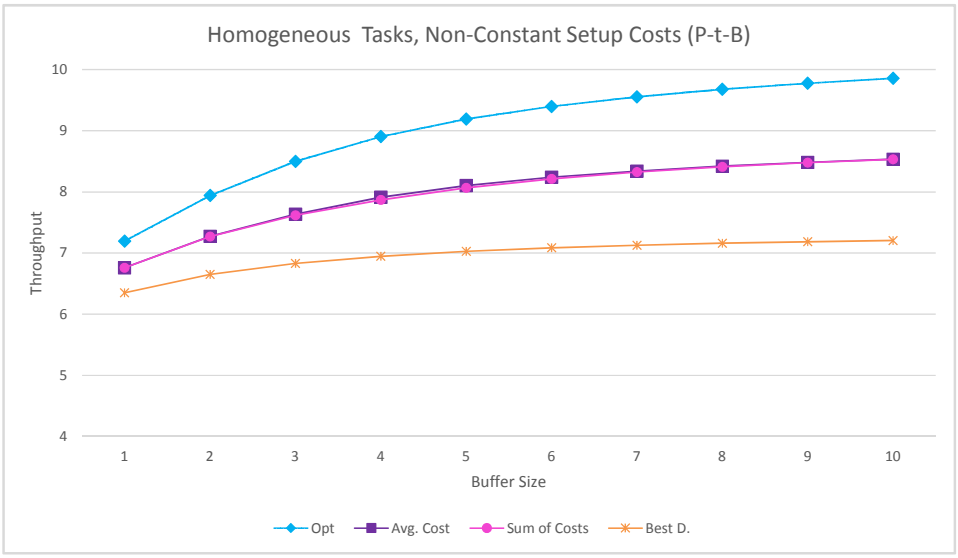

Figure 6: Homogeneous Tasks with Non-Constant Setup Costs (P-t-B)

service rate structures between the systems with homogeneous and non-homogeneous tasks. When the tasks are homogeneous, there is a dominating server that equally dominates the other server at both stations. If the tasks are non-homogeneous with a dominating server, then it is possible that the dominating server dominates the other server to a different degree at each station, possibly being nearly identical to the slow server at one of the stations. In this case, a dedicated policy is more likely to be optimal given a setup cost value, hence the performance of the best dedicated policy improves for systems with non-homogeneous tasks.

Also note that if the buffer size is large (i.e., $B_1 = 10$), the heuristic policies perform similar to each other with an optimality gap around 2%. For smaller buffer sizes (i.e., $B_1 = 1, \ldots, B_1 = 5$), the heuristic policy that uses the *homogenized rates* and the *sum of the costs* performs the best with an optimality gap of 2.5% to 3% whereas other heuristics yield optimality gaps of 19% (*homogenized rates-minimum of the costs*, $B_1 = 1$) to 3% (*homogenized rates-average cost*, $B_1 = 5$). Once again, we observe that a heuristic that uses a larger setup cost value (i.e., one that is more likely

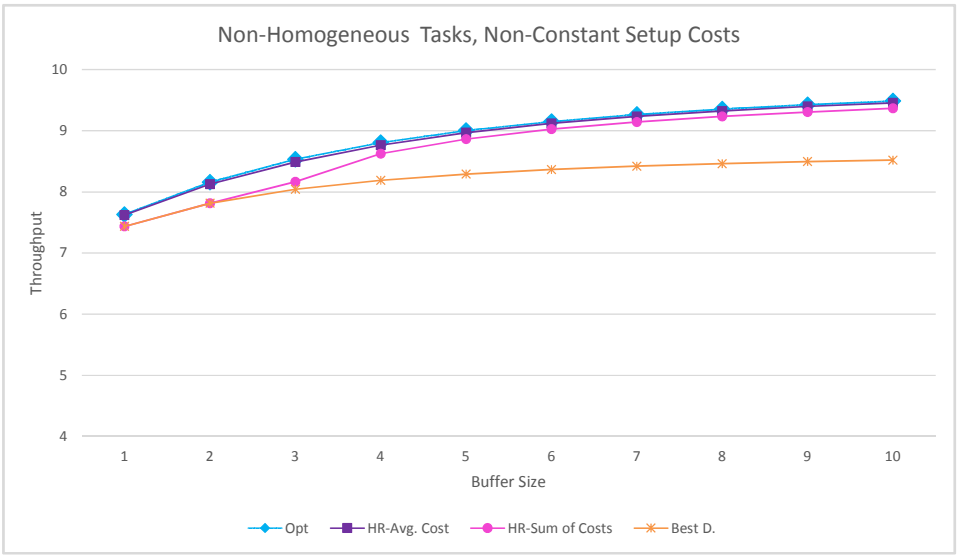

Figure 7: Non-Homogeneous Tasks with Non-Constant Setup Costs (P-t-B)

to yield a dedicated server assignment policy) performs better for small buffer sizes, suggesting that a policy that picks the best performing policy among the heuristic policies and the dedicated policies might perform near-optimal. Figure 7 shows the performances of P-t-B policies for *average cost* and *sum of the costs* in systems with non-homogeneous tasks. All P-t-B heuristics except the heuristic policy that uses the *homogenized rates* and the *sum of the costs* (HR-S. of C.) result in an improvement over the heuristics without the comparison step. The lack of improvement for the *homogenized rates-sum of the costs* heuristic can be explained by the relatively better performance of the heuristic in systems where dedicated policies also perform well. Note that *homogenized rates-sum of the costs* heuristic tends to switch the servers less often compared to the other heuristics due to the higher cost of setups. This counteracts the effects of homogenizing the service rates leading to improved performace. The optimality gap for the remaining P-t-B heuristics is less than 0.5% for all buffer sizes. Hence, choosing the best dedicated policy when it performs better than the heuristic policies results in near-optimal long-run average profit. In Appendix B,

Table 31 shows 95% confidence intervals for the *pick-the-best* policies that correspond to each of the heuristic server assignment policies devised earlier.

4.6 Conclusions

We study systems of tandem queues with flexible non-collaborative servers and setup costs. For tandem systems of N stations with general service requirements and flexible servers, we show that a dedicated server assignment policy is optimal if there is a distinct server that is the fastest at each station or if the setup costs are large enough. Furthermore, we completely characterize the optimal server assignment policy in two station systems with a finite buffer, homogeneous tasks, and flexible servers when server reassignments come at a cost. Our results show that the optimal policy is of double threshold type with symmetric thresholds that are dependent on the magnitude of the setup cost. We also conclude that benefits of cross-training diminish as the setup cost increases. Finally we show that cross-training becomes beneficial regardless of the magnitude of the setup costs as the buffer size grows.

For systems with non-homogeneous tasks and/or non-constant setup costs, heuristic server assignment policies are developed. Numerical results show that the dedicated server assignment policies can be near-optimal in systems with non-homogeneous tasks if the buffer size is small and the setup costs are high, which is consistent with our analytical results for systems with homogeneous tasks. When the tasks are homogeneous and the setup costs are dependent on the server assignment used, the dedicated server assignment policies perform poorly for all buffer sizes, and are outperformed by all the dynamic server assignment heuristics we developed. In both settings, the *Pick-the-Best* heuristic that selects the best performing policy between the dedicated policies and our homogenized heuristics yields near-optimal long-run average profits for all setup cost values, regardless of the buffer size.

CHAPTER V

PROCESSOR SHARING IN NON-COLLABORATIVE NETWORKS

5.1 Introduction

This chapter aims to analyze a general class of finite buffered systems with flexible servers. The approach involves using a processor sharing scheme to design dynamic server allocation policies for these systems, and in the process we establish new results for processor sharing queues.

We consider a general queueing network with N stations and M servers, where $M \leq N$, and the buffers between stations can be finite or infinite. Servers are flexible but unable to collaborate at a station due to physical limitations and/or insufficient tooling. Thus, each server is cross-trained to work at multiple stations, but there can be at most one server working at each station at any given time. There are $K \leq N$ job types in the system. We assume there is infinite number of jobs of each type available to be processed and there is infinite room for completed jobs.

We allow for general routing and topology, and assume that service requirements at station j are independent and identically distributed random variables for $j \in \{1, \ldots, N\}$. Each server $i \in \{1, \ldots, M\}$ works at a rate μ_{ij} at each station $j \in \{1, \ldots, N\}$, thus server i is trained to work at station j if $\mu_{ij} > 0$. Without loss of generality, we also assume that the mean service requirement is one at all stations. Furthermore, servers can be reassigned to different stations at no cost and switching times are negligible. We let $B_j \geq 0, j \in \{1, \ldots, N-1\}$, denote the size of the buffer between stations j and $j+1$. If the buffers are finite, we assume that the system operates under manufacturing blocking.

The remainder of this chapter is organized as follows. In Section 5.2, we provide a linear program (LP) that yields an upper bound on the achievable throughput in systems with infinite buffers, and identify the processor sharing policies that yield the optimal solution to the linear program for tandem lines. In Section 5.3, we introduce a class of timed round-robin policies that approximate the processor sharing scheme and evaluate their performance in systems with finite buffers and various topologies. Finally, in Section 5.4, we provide our conclusions.

5.2 Processor Sharing in Networks of Queues

In this section, we focus our attention to queueing networks with infinite buffers and non-collaborative flexible servers. We allow for general topology and routing probabilities in our formulation, however most of our results address the tandem case.

For systems with infinite buffers, we assume, without loss of generality, that each job type is stored at a different buffer at a station. In a network with K job types, the type of a job is determined by the buffer at which it enters the network. We let $\{j_1, j_2, \ldots, j_K\}$ be the entry stations for these job types and let λ_k, $k \in \{1, \ldots, K\}$, denote the rate at which type k jobs enter the network. A job that completes service at station l is routed to station j with probability p_{lj} for all pairs of stations $l, j \in \{1, \ldots, N\}$, and leaves the system with probability $1 - \sum_{j=1}^{N} p_{lj}$. We let P be the routing matrix with (l, j) entry p_{lj}, and assume the matrix $(I - P)$ is invertible, where I is the $N \times N$ identity matrix. Thus, every job eventually leaves the network.

For each job type k, we have the traffic equations

$$\lambda_{jk} = \lambda_k \mathbb{1}_{\{j_k = j\}} + \sum_{l=1}^{N} p_{lj} \lambda_{lk}, \qquad \forall j \in \{1, \ldots, N\}, \tag{26}$$

where λ_{jk} is the rate of arrivals for type k jobs at station j, and $\mathbb{1}_{\{\cdot\}}$ is the indicator function. Let α_{jk}, $j \in \{1, \ldots, N\}, k \in \{1, \ldots, K\}$, be the solutions to the traffic equations when $\lambda_k = 1$ for all $k \in \{1, \ldots, K\}$. We formulate the following linear

program (LP) that has a similar structure to the capacity LP given for collaborative queueing networks with a single arrival process by Andradóttir et al. [12]:

$$\max \sum_{k=1}^{K} \lambda_k$$

$$\text{s.t.} \sum_{i=1}^{M} \delta_{ijk}\mu_{ij} \geq \lambda_k \alpha_{jk}, \qquad \forall j \in \{1,\ldots,N\}, k \in \{1,\ldots,K\}; \tag{27}$$

$$\sum_{j=1}^{N}\sum_{k=1}^{K} \delta_{ijk} \leq 1, \qquad \forall i \in \{1,\ldots,M\}; \tag{28}$$

$$\sum_{i=1}^{M}\sum_{k=1}^{K} \delta_{ijk} \leq 1, \qquad \forall j \in \{1,\ldots,N\}; \tag{29}$$

$$\delta_{ijk} \geq 0, \qquad \forall i \in \{1,\ldots,M\}, j \in \{1,\ldots,N\}, k \in \{1,\ldots,K\}, \tag{30}$$

where the decision variables λ_k for $k \in \{1,\ldots,K\}$, are the rates each job type enter the network and the decision variables $\delta_{ijk} \geq 0$ for $i \in \{1,\ldots,M\}, j \in \{1,\ldots,N\}, k \in \{1,\ldots,K\}$, are the long-run average fractions of time server i is assigned to station j to serve type k jobs. Thus $\delta_{ij} = \sum_{k=1}^{K} \delta_{ijk}$ is the long-run average fraction of time server i is assigned to station j. Note that the service rates $\mu_{ij}, i \in \{1\ldots,M\}, j \in \{1,\ldots,N\}$, and the routing probabilities $p_{lj}, l, j \in \{1,\ldots,N\}$, could also be defined to depend on the type of the job that is being processed and routed (i.e., μ_{ijk} and p_{ljk} can be defined for each job type k) without changing the structure of the LP. Thus, our formulation can be applied to more general systems.

Constraint (27) ensures the stability of the network and constraint (28) prevents overallocation. Note that without constraint (29), the LP can be seen as an adaptation of the capacity LP in [12] to networks with multiple independent arrival processes, in the sense that it determines the maximal total flow $\sum_{k=1}^{K} \lambda_k$ into the network for which all queues are stable. Let λ_k^* and δ_{ijk}^*, for $i \in \{1,\ldots,M\}, j \in \{1,\ldots,N\}, k \in \{1,\ldots,K\}$, be an optimal solution to the LP. In a stable network with infinite buffers, $\sum_{k=1}^{K} \lambda_k^*$ is also the maximum throughput that can be achieved. Note that this interpretation remains true in systems with no arrivals where there are infinitely

many jobs of each type ready to be processed at their respective entry stations. The addition of constraints (29) makes sure that the total server-time assigned to a station cannot exceed one, which is the case in non-collaborative systems. Thus, the adapted capacity LP yields an upper bound on the maximum achievable long-run average throughout for a non-collaborative network.

The solution to the capacity LP can be interpreted to correspond to a processor sharing policy where each server i devotes a δ_{ij} proportion of her service capacity to station j. In particular, we call a processor sharing policy *egalitarian* if a shared server equally distributes her capacity among nodes, so that station j is served by server i at a rate of $\frac{\mu_{ij}}{N}$. We say a processor sharing policy is *non-egalitarian*, if the server prioritizes stations and allocates her capacity according to a set of priority coefficients $\{w_{ij} : i \in \{1,\dots,M\}, j \in \{1,\dots,N\}\}$, so that station $j \in \{j_1, j_2, \dots, j_k\}$ is served by server i at the rate of $\frac{w_{ij}\mu_{ij}}{\sum_{j=1}^{N} w_{ij}}$. The following result focuses on tandem queueing networks with a single job type, and shows that if the tasks are homogeneous so that the service rates are only dependent on the server (i.e., $\mu_{ij} = \mu_i$), then an egalitarian processor sharing policy provides the optimal solution to the adapted LP.

Proposition 11. *The capacity LP for a non-collaborative tandem queueing network with N homogeneous tasks, $M \leq N$ servers, and a single type of jobs at the first station ($K = 1$) yields $\lambda_1^* = \frac{\sum_{i=1}^{M} \mu_i}{N}$. Moreover, the egalitarian processor sharing scheme provides a possible allocation with $\delta_{ij1}^* = \frac{1}{N}$, for all $i \in \{1,\dots,M\}, j \in \{1,\dots,N\}$.*

Proof. It is easy to verify that $\delta_{ij1} = \frac{1}{N}$ is a feasible solution when $M \leq N$. For a tandem line, we have $p_{lj} = 1$ for all pairs of j, l such that $j = l + 1$, $j \in \{1,\dots,N\}$, and $p_{lj} = 0$ otherwise. The traffic equations yield $\alpha_{j1} = 1$ for all $j \in \{1,\dots,N\}$, and the constraint (27) becomes

$$\sum_{i=1}^{M} \delta_{ij1}\mu_i \geq \lambda_1 \qquad \forall j \in \{1,\dots,N\}.$$

Therefore, for any optimal solution $\{\delta^*_{ij1} : i \in \{1,\ldots,M\}, j \in \{1,\ldots,N\}\}$ and λ^*_1, we must have $\lambda^*_1 = \min_j \left\{ \sum_{i=1}^M \delta^*_{ij1}\mu_i \right\}$, where station j is considered a bottleneck if $\lambda^*_1 = \sum_{i=1}^M \delta^*_{ij1}\mu_i$.

We first show that all stations must be bottlenecks at the optimal solution to the capacity LP. Suppose there exists a non-bottleneck station l so that

$$\sum_{i=1}^M \delta^*_{il1}\mu_i > \min_j \left\{ \sum_{i=1}^M \delta^*_{ij1}\mu_i \right\} = \lambda^*. \tag{31}$$

If there is a bottleneck station j^* such that $\sum_{i=1}^M \delta^*_{ij^*1} < 1$, then for any server with $\delta^*_{il1} > 0$, we can choose an $\epsilon > 0$ and assign $\delta'_{il1} = \delta^*_{il1} - \epsilon$, $\delta'_{ij^*1} = \delta^*_{ij^*1} + \epsilon$ maintaining a feasible solution to the capacity LP. Thus, $\sum_{i=1}^M \delta'_{ij^*1}\mu_i > \sum_{i=1}^M \delta^*_{ij^*1}\mu_i = \lambda^*_1$ and the bottleneck can be improved, contradicting λ^*_1 is optimal.

Now assume that for all bottleneck stations j^* the constraint (29) is tight, so we have $\sum_{i=1}^M \delta^*_{ij^*1} = 1$. Then, there must exist a pair of servers i_1, i_2 such that $\mu_{i_1} > \mu_{i_2}$ and $\delta^*_{i_1 l1} > 0$, $\delta^*_{i_2 l1} < 1$, since otherwise only the slowest server must be assigned to the non-bottleneck station l, contradicting our assumptions that (31) holds and $\sum_{i=1}^M \delta^*_{ij^*1} = 1$. Furthermore among such pairs of servers i_1, i_2, there must exist one such that $\delta_{i_1 j^*1} < 1, \delta_{i_2 j^*1} > 0$. Note that $\delta_{i_1 j1} < 1$ is implied by $\delta_{i_1 l1} > 0$ and the constraint (28), and having $\delta_{i_2 j1} = 0$ for all such pairs contradicts (31). Thus, we can pick an $\epsilon > 0$ and set $\delta'_{i_1 j^*1} = \delta^*_{i_1 j^*1} + \epsilon$, $\delta'_{i_2 j^*1} = \delta^*_{i_2 j^*1} - \epsilon$, $\delta'_{i_1 l1} = \delta^*_{i_1 l1} - \epsilon$, and $\delta'_{i_2 l1} = \delta^*_{i_2 l1} + \epsilon$, while maintaining a feasible solution to the capacity LP. Since $\mu_{i_1} > \mu_{i_2}$, we have $\sum_{i=1}^M \delta'_{ij^*1}\mu_i > \sum_{i=1}^M \delta^*_{ij^*1}\mu_i = \lambda^*_1$ and λ^*_1 cannot be optimal. Therefore, we must have no non-bottleneck stations at the optimal solution, and we have $\lambda^*_1 = \sum_{i=1}^M \delta^*_{ij1}\mu_i$ for all $j \in \{1,\ldots,N\}$.

Second, we show that there cannot be any idle servers in the optimal solution. Suppose there exists a server i' that is idled for a proportion of the time. Then, there must also exist a station j' with $\sum_i \delta^*_{ij'1} < 1$ since $M \leq N$. Therefore, we can increase $\delta^*_{i'j'1}$ by some $\epsilon > 0$ while maintaining feasibility and optimality, so that the station j'

is no longer a bottleneck, which contradicts that every station must be a bottleneck in an optimal solution.

We showed that at the optimal solution, we must have the sets of constraints (27) and (28) tight. Summing (27) over all stations $j \in \{1, \ldots, N\}$, we get

$$\sum_{j=1}^{N} \sum_{i=1}^{M} \delta_{ij1}^{*} \mu_i = \sum_{i=1}^{M} \mu_i \sum_{j=1}^{N} \delta_{ij1}^{*} = \sum_{i=1}^{M} \mu_i = N\lambda_1^{*}.$$

Therefore, $\lambda_1^{*} = \frac{\sum_{i=1}^{M} \mu_i}{N}$, completing the proof. □

Note that processor sharing is an idealized server assignment scheme and assumes a server can simultaneously serve multiple stations at a constant rate (without any switching), and a station can get simultaneously served by multiple servers. In a non-collaborative network with dynamically assigned servers, these assumptions are not valid. However, processor sharing provides a direct interpretation for the upper bound yielded by the capacity LP.

It also follows from Proposition 11 that allowing collaboration of the servers does not improve the capacity upper bound for tandem network if the tasks are homogeneous.

Corollary 3. *For tandem lines with homogeneous tasks and $M \leq N$, the non-collaborative capacity LP has the same optimal objective function value as the collaborative one.*

Proof. Corollary 2 of Andradóttir et al. [12] show that in a collaborative tandem line with N stations, M servers, infinite buffers, and homogeneous tasks, the optimal objective function value for the collaborative capacity LP (without the set of constraints (29)) is $\lambda_1^{*} = \frac{\sum_{i=1}^{M} \mu_i}{N}$. Thus adding the new set of constraints (29) does not change the maximal capacity. □

If the tasks in a tandem queueing network are non-homogeneous, the egalitarian processor sharing scheme no longer yields the capacity upper bound. The following

result shows that if the servers are generalists and the service rates are dependent both on the server and the station, so that $\mu_{ij} = \mu_i\gamma_j$, then the structure of the optimal policy depends on whether there is a bottleneck station or not. Before we present the result, let us define a non-idling non-egalitarian processor sharing scheme for a tandem line with two stations and two servers. Under the non-egalitarian processor sharing policy π^δ, server 1 dedicates the proportion δ of her capacity to station 1 and the remaining $1-\delta$ proportion to station 2. Similarly, server 2 dedicates a δ proportion of her capacity to station 2 and the remaining $1-\delta$ proportion to station 1. (Note that since π^δ is non-idling, the capacity of server 1 dedicated to station 1 must be equal to the capacity of server 2 dedicated to station 2.) Also note that if $\mu_1 = \mu_2$, any non-idling policy results in the same objective function value and is optimal. We assume $\mu_1 > \mu_2$, without loss of generality, and define the following quantity

$$\delta^* = \frac{\mu_1\gamma_2 - \mu_2\gamma_1}{(\mu_1 - \mu_2)(\gamma_1 + \gamma_2)} \in \mathbb{R},$$

which can be used as the proportion parameter for a non-egalitarian processor sharing policy when it is in $[0, 1]$.

Proposition 12. *In non-collaborative systems with infinite buffers, two stations, a single type of jobs ($K = 1$), and two generalists servers with rates $\mu_{ij} = \mu_i\gamma_j$ for $i, j \in \{1, 2\}$,*

i. If $\mu_1\gamma_1 \leq \mu_2\gamma_2$, then the dedicated server assignment policy that keeps server 1 (server 2) at station 1 (station 2) at all times is optimal.

ii. If $\mu_1\gamma_2 \leq \mu_2\gamma_1$, then the dedicated server assignment policy that keeps server 2 (server 1) at station 1 (station 2) at all times is optimal.

iii. If $\mu_1\gamma_1 \geq \mu_2\gamma_2$ and $\mu_1\gamma_2 \geq \mu_2\gamma_1$, then the non-egalitarian processor sharing policy π^{δ^*} yields an optimal solution to the capacity LP.

Proof. Note that a solution to the capacity LP cannot be optimal if it idles the faster server, since λ_1^* can be improved by assigning the faster server to the bottleneck station or stations for the duration of time she was idled, and idling the slower server instead if necessary (due to the constraints (29) in the capacity LP).

If $\mu_1\gamma_1 \leq \mu_2\gamma_2$, we have $\sum_{i=1}^{M} \delta_{i11}^* \mu_i\gamma_1 \leq \sum_{i=1}^{M} \delta_{i21}^* \mu_i\gamma_2$ for any optimal solution $\{\delta_{ij1}^*\}$, λ_1^*, and the first station is a bottleneck at the optimal solution with $\sum_{i=1}^{M} \delta_{i11}^* \mu_i\gamma_1 = \lambda_1^*$. Therefore, the dedicated policy that keeps server 1 at the first station maximizes λ_1, and is optimal. Similarly, if $\mu_1\gamma_2 \leq \mu_2\gamma_1$, we have $\sum_{i=1}^{M} \delta_{i21}^* \mu_i\gamma_2 \leq \sum_{i=1}^{M} \delta_{i11}^* \mu_i\gamma_1$ for any optimal solution $\{\delta_{ij1}^*\}$, λ_1^*, since server 1 is not idled. Thus, the second station is a bottleneck at the optimal solution and we have $\sum_{i=1}^{M} \delta_{i21}^* \mu_i\gamma_2 = \lambda_1^*$, implying the dedicated policy that keeps server 1 at the second station maximizes λ_1, and is optimal.

For the case with $\mu_1\gamma_1 \geq \mu_2\gamma_2$, we first show that at the optimal solution we must have $\sum_{i=1}^{M} \delta_{i11}^* \mu_i\gamma_1 = \sum_{i=1}^{M} \delta_{i21}^* \mu_i\gamma_2 = \lambda_1^*$. Assume we have

$$\sum_{i=1}^{M} \delta_{i11}^* \mu_i\gamma_1 < \sum_{i=1}^{M} \delta_{i21}^* \mu_i\gamma_2. \tag{32}$$

Then, we must have $\delta_{111}^* < 1$, because $\delta_{111}^* = 1$ implies $\delta_{211}^* = \delta_{121}^* = 0$, contradicting (32). We also must have $\delta_{121}^* > 0$ since server 1 is not idled. Therefore, there exists an $\epsilon > 0$ and $\epsilon \geq \epsilon_2 \geq 0$ such that $\delta'_{111} = \delta_{111}^* + \epsilon$, $\delta'_{121} = \delta_{121}^* - \epsilon$, $\delta'_{211} = \delta_{211}^* - \epsilon_2$, and $\delta'_{221} = \delta_{22}^*$ yield a feasible solution to the capacity LP and $\lambda'_1 = \sum_{i=1}^{M} \delta'_{i11} \mu_i\gamma_1 < \sum_{i=1}^{M} \delta'_{i21} \mu_i\gamma_2$. Furthermore, we have $\lambda'_1 = \sum_{i=1}^{M} \delta'_{i11} \mu_i\gamma_1 > \sum_{i=1}^{M} \delta_{i11}^* \mu_i\gamma_1 = \lambda_1^*$, thus λ_1^* cannot be optimal. Similarly, if we have

$$\sum_{i=1}^{M} \delta_{i11}^* \mu_i\gamma_1 > \sum_{i=1}^{M} \delta_{i21}^* \mu_i\gamma_2, \tag{33}$$

we must have $\delta_{121} < 1$ since $\delta_{121} = 1$ implies $\delta_{111} = \delta_{221} = 0$, contradicting (33). We also have $\delta_{111} > 0$ since server 1 is not idled. Then, there exists $\epsilon > 0$ and $\epsilon \geq \epsilon_2 \geq 0$ such that $\delta'_{111} = \delta_{111}^* - \epsilon$, $\delta'_{121} = \delta_{121}^* + \epsilon$, $\delta'_{221} = \delta_{221}^* - \epsilon_2$, and $\delta'_{211} = \delta_{211}^*$ yields

a feasible solution and $\sum_{i=1}^{M} \delta'_{i11}\mu_i\gamma_1 > \sum_{i=1}^{M} \delta'_{i21}\mu_i\gamma_2 = \lambda'_1$. Furthermore, we have $\lambda^*_1 = \sum_{i=1}^{M} \delta^*_{i21}\mu_i\gamma_2 < \sum_{i=1}^{M} \delta'_{i21}\mu_i\gamma_2 = \lambda'_1$, thus λ^*_1 cannot be optimal.

Since we must have $\sum_{i=1}^{M} \delta^*_{i11}\mu_i\gamma_1 = \sum_{i=1}^{M} \delta^*_{i21}\mu_i\gamma_2 = \lambda^*_1$ at the optimal solution and all stations are bottlenecks, it follows from the proof of Proposition 11 that the optimal server does not idle any server. Thus, at any feasible optimal solution we have $\delta^*_{121} = \delta^*_{211} = 1 - \delta^*_{111}$ and $\delta^*_{221} = \delta^*_{111}$. Solving for δ^*_{111}, we get

$$\delta^*_{111} = \frac{\mu_1\gamma_2 - \mu_2\gamma_1}{(\mu_1 - \mu_2)(\gamma_1 + \gamma_2)}.$$

Note that $\delta^*_{111} \geq 0$ since $\mu_1\gamma_2 \geq \mu_2\gamma_1$, and $\delta^*_{111} \leq 1$ since $\mu_1\gamma_1 \geq \mu_2\gamma_2$, thus the solution $\{\delta^*_{ij1}\}$, λ^*_1 is feasible and optimal. □

Remark 6. *Note that the policy given by Proposition 12 does not need to be unique. Furthermore, if the service rates are only dependent on the station, so that $\mu_{ij} = \mu\gamma_j$ for $i, j \in \{1, 2\}$, every non-idling server assignment policy results in average service rates $\mu\gamma_1$ in station 1 and $\mu\gamma_2$ in station 2, yielding a long-run average throughput of $\min\{\mu\gamma_1, \mu\gamma_2\}$. Thus, any non-idling server assignment policy is optimal.*

In a non-collaborative two-station system, if there is no bottleneck station (i.e., $\mu_1\gamma_1 \geq \mu_2\gamma_2$ and $\mu_1\gamma_2 \geq \mu_2\gamma_1$), we have $\delta^* \in [0, 1]$, and the optimal processor sharing policy π^{δ^*} yields equal service rates at both stations without voluntarily idling the servers. If there is a bottleneck station so that the service at that station is slower even when the fastest server (i.e., server 1) is assigned, then it is optimal to keep the fastest server dedicated at the bottleneck. On the other hand, in a collaborative system, it is possible to increase the service rate at a bottleneck station until it is no longer a bottleneck by assigning both servers to the same station. Thus, the optimal processor sharing policy cannot result in a single bottleneck station for collaborative systems.

Proposition 13. *Allowing collaboration improves the long-run average throughput in two-station networks with two generalist servers, infinite buffers, and a single type*

of jobs at the first station, if and only if the non-collaborative optimal policy yields a single bottleneck station.

Proof. Suppose we have $\mu_1\gamma_1 < \mu_2\gamma_2$ and the policy that keeps server 1 at station 1 and server 2 at station 2 at all times is optimal. The corresponding optimal solution to the capacity LP yields $\lambda_1^* = \sum_{i=1}^{M} \delta_{i1}^* \mu_i \gamma_1 < \sum_{i=1}^{M} \delta_{i2}^* \mu_i \gamma_2$ with $\delta_{11}^* = 1$, $\delta_{12}^* = 0$,$\delta_{21}^* = 0$, and $\delta_{22}^* = 1$. If the servers are allowed to collaborate, we can pick an $\epsilon > 0$ such that server 2 can be moved to station 1 for an ϵ proportion of time to work together with server 1 without changing the bottleneck station. Then, the new server assignment with $\delta_{11}' = 1$, $\delta_{12}' = 0$, $\delta_{21}' = \epsilon$, and $\delta_{22}' = 1 - \epsilon$ yields $\lambda_1' = \mu_1\gamma_1 + \epsilon(\mu_2\gamma_1) > \mu_1\gamma_1 = \lambda_1^*$, improving the long-run average throughput.

A similar argument can be made for the case with $\mu_1\gamma_2 < \mu_2\gamma_1$, since server 2 can be moved to station 2 to work together with server 1 for a proportion of time while keeping station 2 as the bottleneck station. □

Remark 7. *Corollary 3 states that if the tasks are homogeneous, collaboration is not beneficial in tandem lines with infinite buffers. By contrast, collaboration can be valuable if the tasks are non-homogeneous.*

5.3 Non-Collaborative Networks with Finite Buffers

The capacity LP yields an upper bound on the achievable long-run average throughput since any non-collaborative server assignment policy will results in a feasible solution to the LP. Our results in Section 2 show that this upper bound can be achieved by a processor sharing scheme when buffers are infinite. Processor sharing is a suitable model for systems where multiple jobs can receive service simultaneously at fractional capacities (e.g., communication networks with shared bandwidth). However, it may not be directly implementable in manufacturing systems where servers are non-collaborative and utilized at different tasks through dynamic server allocation. Thus, it is unclear whether the upper bound yielded by the capacity LP is attainable

in these systems. Furthermore, the performance of the processor sharing scheme in systems with finite buffers is unknown.

In this section, we analyze the performance of processor sharing policies in finite buffered systems and develop server assignment policies for non-collaborative networks. In Section 5.3.1, we show that the processor sharing policies are asymptotically optimal in systems with two stations and finite buffers and modify the processor sharing scheme introduced in the previous section to improve its performance in systems with finite buffers. Then we introduce a class of round-robin server allocation policies for systems with non-collaborative servers and finite buffers, and show that these policies approximate processor sharing in systems with two stations. For larger queueing networks, we evaluate the performance of the round-robin policies through numerical experiments in Section 5.3.2.

5.3.1 Processor Sharing in Systems with Finite Buffers

The throughput-optimal server assignment policy and the corresponding long-run average throughput for a tandem Markovian two-station system with a finite buffer and homogeneous tasks is characterized in Chapter 3. Let $T^*(B_1)$ denote the optimal throughput in such a system and $T^{\pi^{ps}}(B_1)$ denote the throughput achieved by the egalitarian processing sharing policy in the same system. Using equation (2) in Chapter 3, we have

$$T^*(B_1) = \frac{\sum_{i=1}^{s^*-1} \mu_1^{B_1+3-s^*+i}\mu_2^{s^*-i} + \sum_{i=0}^{B_1+2-s^*} \mu_1^{B_1+3-i}\mu_2^{i}}{\sum_{i=1}^{s^*} \mu_1^{B_1+2-s^*+i}\mu_2^{s^*-i} + \sum_{i=0}^{B_1+2-s^*} \mu_1^{B_1+2-i}\mu_2^{i}} \quad \text{where } s^* = \lfloor \frac{B_1+3}{2} \rfloor,$$

and

$$T^{\pi^{ps}}(B_1) = \frac{(B_1+2)(\mu_1+\mu_2)}{(B_1+3)2}. \tag{34}$$

The following result shows that for Markovian tandem systems with two stations and homogeneous tasks, the optimal policy for finite-buffered systems asymptotically achieves the processor sharing bound.

Proposition 14. *The optimal long-run average throughput for a Markovian two-station non-collaborative tandem system with a finite buffer and homogeneous tasks converges to the throughput of the egalitarian processor sharing policy as buffer size grows to infinity.*

Proof. Note that we have $T^*(B_1) \geq T^{\pi^{ps}}(B_1)$ since $T^*(B_1)$ is the optimal throughput. Furthermore, $\lim_{B_1 \to \infty} T^*(B_1) = \lim_{B_1 \to \infty} T^{\pi^{ps}}(B_1) = \frac{\mu_1 + \mu_2}{2}$, and thus

$$\lim_{B_1 \to \infty} [T^*(B_1) - T^{\pi^{ps}}(B_1)] = 0,$$

completing the proof. □

Note that the processor sharing scheme we introduced for systems with infinite buffers has servers distribute their service capacities among all stations regardless of whether there is a job to be served at a station or not. Thus, when there are starved stations, the capacity assigned to these stations is idled until a job arrives from another station. In a network with finite buffers, idling the servers in this manner is disadvantageous since it reduces the long-run average throughput by partially idling the faster servers that are shared by starved or blocked stations, while they can be utilized to increase the service rates in the busy stations. To improve the performance of processor sharing policies in systems with finite buffers and homogeneous tasks, we modify the processor sharing scheme discussed in Section 2 as follows. Without loss of generality, we assume the service rates μ_i are ordered so that the lowest indexed server is the fastest (i.e., $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_M$). Suppose there are n stations $\{j_1, j_2, \ldots, j_n\} \subset \{1, \ldots, N\}$ with unfinished jobs present and the remaining stations are starved or blocked. If $n < M$, the modified processor sharing scheme has the fastest n servers with rates $\{\mu_1, \mu_2, \ldots, \mu_n\}$ shared by the stations $\{j_1, j_2, \ldots, j_n\}$, and idles the remaining $M - n$ servers. (If $n \geq M$, then all M servers are shared by the n busy stations.) As in systems with infinite buffers, we consider a processor

sharing policy to be egalitarian if a shared server allocates her service capacity equally among the stations served by her, and to be non-egalitarian otherwise.

For a Markovian two-station tandem system with homogeneous tasks and a finite buffer of size B_1, the modified egalitarian processor sharing policy results in a birth-death process with birth and death rates of $\frac{\mu_1+\mu_2}{2}$ when both stations are busy, birth rate of μ_1 when the second station is starved, and death rate of μ_1 when the first station is blocked. Thus, the long-run average throughput attained by the modified processor sharing policy can be computed as

$$T^{\pi^{mps}}(B_1) = \frac{(B_1+2)\mu_1(\mu_1+\mu_2)}{2(B_1+2)\mu_1+2\mu_2}.$$

Furthermore, using (34) and Proposition 14, we have $\lim_{B_1\to\infty} T^{\pi^{mps}}(B_1) = \lim_{B_1\to\infty} T^*(B_1) = \frac{\mu_1+\mu_2}{2}$ and $T^{\pi^{mps}}(B_1) > T^{\pi^{ps}}(B_1)$ for all finite buffer sizes B_1. Note we also have $T^*(0) = T^{\pi^{mps}}(0) = \frac{\mu_1(\mu_1+\mu_2)}{2\mu_1+\mu_2}$, and the modified processor sharing policy yields the optimal long-run average throughput when the buffer size is zero.

For systems where processor sharing is not implementable, we propose the following class of round-robin policies to approximate the modified processor sharing scheme. For a queueing network with N servers and M stations, consider a timed round-robin policy π^Q that rotates the servers among all tasks according to a deterministic clock. We let $\mathscr{A}$ denote the set of all server assignments. Each server assignment $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_M)$ is a vector where σ_i is the station the server i is assigned to and $Q = \{q_{V,\sigma} : \sigma \in \mathscr{A}, V \subset \{1,\ldots,N\}\}$ is a set, where $q_{V,\sigma}$ indicates the duration of time the server assignment σ should be employed in the rotation for the set V of busy stations. In particular, for a set of busy stations V, the policy π^Q starts with an initial server assignment σ and keeps this server assignment for $q_{V,\sigma}$ units of time before moving to the next server assignment. We say a server assignment is eligible for the set of busy stations V, if it assigns the slowest servers to the starved or blocked stations. Let $\mathscr{A}_V$ denote the set of eligible assignments for the set of busy stations V. Then, the policy π^Q has $q_{V,\sigma} = 0$ for all $\sigma \notin \mathscr{A}_V$, $V \subset \{1,\ldots,N\}$, and

given the set of busy stations V, the timed round-robin policy π^Q only includes the fastest $|V|$ servers in the rotation.

We define the capacity LP for a subset of stations V to be

$$\begin{aligned}
\max \ & \sum_{k=1}^{K} \lambda_k \\
\text{s.t.} \ & \sum_{i=1}^{M} \delta_{ijk}\mu_i \geq \lambda_k \alpha_{jk}, \qquad \forall j \in V, k \in \{1,\ldots,K\}; \\
& \sum_{j\in V}\sum_{k=1}^{K} \delta_{ijk} \leq 1, \qquad \forall i \in \{1,\ldots,M\}; \\
& \sum_{i=1}^{M}\sum_{k=1}^{K} \delta_{ijk} \leq 1, \qquad \forall j \in V; \\
& \delta_{ijk} = 0, \qquad \forall i \notin \{1,\ldots,|V|\}, j \in V, k \in \{1,\ldots,K\}, \\
& \delta_{ijk} \geq 0, \qquad \forall i \in \{1,\ldots,\min\{|V|,M\}\}, j \in V, k \in \{1,\ldots,K\}.
\end{aligned}$$

Then the capacity LP can be used to compute the parameters $q_{V,\sigma}$ for a timed round-robin policy for a given queueing network with infinite buffers as follows.

(i) For the set of busy stations V, solve the capacity LP to obtain the allocation variables δ^*_{ij}.

(ii) Compute a time proportion $q'_{V,\sigma}$ for each server assignment σ, so that $\delta^*_{ij} \leq \sum_{\sigma\in A_{V,ij}} q'_{V,\sigma}$, for all $i \in \{1,\ldots,M\}, j \in \{1,\ldots,M\}$, and $\sum_{\sigma\in A_V} q'_{V,\sigma} \leq 1$ where $A_{V,ij} \subset \mathscr{A}_V$ is the set of eligible server assignments that have server i at station j. (Note that a solution to the set of inequalities given in this step can be found solving a linear program.)

(iii) Choose a time increment ϵ and assign $q_{V,\sigma} = \epsilon q'_{V,\sigma}$.

(iv) When the set of active stations V changes, re-solve the capacity LP to update the parameters $q_{V,\sigma}$ for all $\sigma \in \mathscr{A}_V$.

The server assignment algorithm given in (i) – (iv) above uses a modified processor sharing policy to construct a round-round robin policy for non-collaborative networks with finite buffers. Note that a similar algorithm with $V = \{1, \ldots, N\}$ can be used to construct a round-robin policy for networks with infinite buffers. To demonstrate how well the resulting round-robin policy can approximate a processor sharing scheme, we consider a tandem queueing network with a single class of jobs at the first station ($K = 1$) and $M = N$ servers, finite buffers, and homogeneous tasks. For the set of busy stations $V = \{j_1, j_2, \ldots, j_{|V|}\}$ (and the non-busy stations $\{j_{|V|+1}, \ldots, j_N\}$), the capacity LP yields $\delta_{ij} = \frac{1}{|V|}$ for $j \in V$, $i \in \{1, \ldots, |V|\}$ by Proposition 11. Thus, we can assign $q'_{V,\sigma} = \frac{1}{|V|}$ for the server assignments $\sigma_k = (\sigma_{k1}, \ldots, \sigma_{kN})$ such that

$$\sigma_{ki} = \begin{cases} j_{i+k} & \text{for } i \in \{1, \ldots, |V| - k\}, \\ j_{i+k-|V|} & \text{for } i \in \{|V| - k + 1, \ldots, |V|\}, \\ j_i & \text{for } i \in \{|V| + 1, \ldots, N\}, \end{cases}$$

for $k \in \{0, \ldots, |V| - 1\}$, and $q'_{V,\sigma} = 0$ for any other assignment in $A_{V,ij}$. The corresponding timed round-robin policy is simple. With every server reassignment, each non-idling server moves to the first downstream station that is busy (unless the server is at the end of the line in which case she moves back to the first active station) and equal time is spent with each server assignment used. Let π^q denote the resulting round-robin policy with $Q = \{q_{V,\sigma} : q_{V,\sigma} = q \text{ if } q'_{V,\sigma} > 0, q_{V,\sigma} = 0, \text{otherwise}\}$, so that servers are reassigned every q time units. Also, let D_n^q, $D_n^{\pi^{mps}}$ denote the departure times of the n^{th} job under π^q and the egalitarian modified processor sharing policy π^{mps}, respectively. The following result shows that the timed round-robin policy π^q for a tandem line of two stations converges to the egalitarian processor sharing scheme as the rotation of the servers becomes more frequent.

Theorem 4. *For non-collaborative tandem systems with two stations, two servers,*

and homogeneous tasks,

$$\lim_{q \to 0} D_n^q = D_n^{\pi^{mps}},$$

and the timed round-robin policy π^q asymptotically achieves the long-run average throughput yielded by egalitarian processor sharing.

Proof. Let $\phi_{j,n}$ denote the service requirement of the n^{th} job at station j for $j \in \{1, 2\}$, $n \in \{1, 2, \dots\}$. Also, let $l^\pi(\phi_{j,n})$ be the time the n^{th} spends at station $j \in \{1, 2\}$ under a policy π. We assume that the system is initially empty and the service requirements follow continuous distributions.

We first show that the departure time of the first job from the system under the round-robin policy π^q converges to the departure time under π^{mps}. The first job enters the system at time zero, and its departure time from the system is the sum of the processing times at the two stations. Thus, we have $D_1^\pi = l^\pi(\phi_{1,1}) + l^\pi(\phi_{2,1})$ under any policy π. The modified processor sharing policy π^{mps} assigns server 1 to the first station while the first job is being processed, hence $l^{\pi^{mps}}(\phi_{1,1}) = \frac{\phi_{1,1}}{\mu_1}$. Once the first job moves to the second station, the second job starts being processed at the first station and both servers are shared under the policy π^{mps}. Thus, the service rates at the both stations are $\frac{\mu_1+\mu_2}{2}$, unless the first station becomes blocked before the departure of the first job from the system. The first job departs the system before the jobs $n \in \{2, \dots, B_1 + 2\}$ finish being served at the first station if

$$\phi_{2,1} < \sum_{i=2}^{B_1+2} \phi_{1,i}, \tag{35}$$

and we have

$$l^{\pi^{mps}}(\phi_{2,1}) = \begin{cases} \frac{2\phi_{2,1}}{\mu_1+\mu_2} & \text{if } \phi_{2,1} < \sum_{i=2}^{B_1+2} \phi_{1,i}, \\ \frac{2(\sum_{i=2}^{B_1+2} \phi_{1,i})}{\mu_1+\mu_2} + \frac{(\phi_{2,1} - \sum_{i=2}^{B_1+2} \phi_{1,i})}{\mu_1} & \text{otherwise.} \end{cases}$$

Therefore, the departure time of the first job under the policy π^{mps} becomes

$$D_1^{\pi^{mps}} = \begin{cases} \frac{\phi_{1,1}}{\mu_1} + \frac{2\phi_{2,1}}{\mu_1+\mu_2} & \text{if } \phi_{2,1} < \sum_{i=2}^{B_1+2} \phi_{1,i}, \\ \frac{\phi_{1,1}}{\mu_1} + \frac{2(\sum_{i=2}^{B_1+2} \phi_{1,i})}{\mu_1+\mu_2} + \frac{(\phi_{2,1}-\sum_{i=2}^{B_1+2} \phi_{1,i})}{\mu_1} & \text{otherwise.} \end{cases} \tag{36}$$

Similarly, the round-robin policy π^q assigns server 1 to the first station while the first job is being processed, and $l^{\pi^q}(\phi_{1,1}) = \frac{\phi_{1,1}}{\mu_1}$. Suppose, without loss of generality, once the first job moves to the second station, the policy π^q initially assigns server 1 to the first station and server 2 to the second station, and alternates the server assignment every q time units. Thus, the service requirements at both stations are depleted by an amount of $q(\mu_1 + \mu_2)$ every $2q$ time units until the first station is blocked or the second station is starved. If the first station is blocked before the first job leaves the second station, then $q(\mu_1 + \mu_2)\lfloor \frac{\sum_{i=2}^{B_1+2} \phi_{1,i}}{q(\mu_1+\mu_2)} \rfloor$ must be the total service requirement served at each station before the first station gets blocked during the next rotation of the servers. We define $R(\phi, q) = \phi - q(\mu_1 + \mu_2)\lfloor \frac{\phi}{q(\mu_1+\mu_2)} \rfloor$ for service requirement ϕ and time increment $q > 0$. Then, the first station becomes blocked before the departure of the first job from the system if

$$R\Big(\sum_{i=2}^{B_1+2} \phi_{1,i}, q\Big)\frac{1}{\mu_1} \leq \Big(\phi_{2,1} - q(\mu_1 + \mu_2)\Big\lfloor \frac{\sum_{i=2}^{B_1+2} \phi_{1,i}}{q(\mu_1 + \mu_2)} \Big\rfloor\Big)\frac{1}{\mu_2} \tag{37}$$

when $R\big(\sum_{i=2}^{B_1+2} \phi_{1,i}, q\big) \leq \mu_1 q$, i.e., the first station becomes blocked while being served by server 1, and if

$$\Big(R\Big(\sum_{i=2}^{B_1+2} \phi_{1,i}, q\Big) - \mu_1 q\Big)\frac{1}{\mu_2} \leq \Big(\phi_{2,1} - q(\mu_1 + \mu_2)\Big\lfloor \frac{\sum_{i=2}^{B_1+2} \phi_{1,i}}{q(\mu_1 + \mu_2)} \Big\rfloor - \mu_2 q\Big)\frac{1}{\mu_1} \tag{38}$$

when $R\big(\sum_{i=2}^{B_1+2} \phi_{1,i}, q\big) > \mu_1 q$, i.e., the first station becomes blocked while being served by server 2. The processing time of the first job at the second station becomes

$$\begin{aligned} l^{\pi^q}(\phi_{2,1}) =& 2q\Big\lfloor \frac{\phi_{2,1}}{q(\mu_1 + \mu_2)} \Big\rfloor + R(\phi_{2,1}, q)\frac{1}{\mu_2}\mathbb{1}_{\{R(\phi_{2,1},q) \leq \mu_2 q\}} \\ &+ \Big(q + (R(\phi_{2,1}, q) - \mu_2 q)\frac{1}{\mu_1}\Big)\mathbb{1}_{\{R(\phi_{2,1},q) > \mu_2 q\}} \end{aligned} \tag{39}$$

if the first job leaves the system before the first station becomes blocked. Otherwise, $l^{\pi^q}(\phi_{2,1})$ can be written as the sum of the time the first job spends at the second station before and after the first station becomes blocked. Let $l_1^{\pi^q}(\phi_{2,1})$ be the time the first job spends in the second station before the first station becomes blocked, and let $l_2^{\pi^q}(\phi_{2,1})$ be the time it spends in the second station after the first station becomes blocked. Then, we have

$$l^{\pi^q}(\phi_{2,1}) = l_1^{\pi^q}(\phi_{2,1}) + l_2^{\pi^q}(\phi_{2,1}), \tag{40}$$

where

$$\begin{aligned} l_1^{\pi^q}(\phi_{2,1}) =& 2q\Big\lfloor \frac{\sum_{i=2}^{B_1+2}\phi_{1,i}}{q(\mu_1+\mu_2)}\Big\rfloor + R\big(\sum_{i=2}^{B_1+2}\phi_{1,i}, q\big)\frac{1}{\mu_1}\mathbb{1}_{\{R(\sum_{i=2}^{B_1+2}\phi_{1,i},q)\le\mu_1 q\}} \\ &+ \Big(q + \big(R\big(\sum_{i=2}^{B_1+2}\phi_{1,i}, q\big) - \mu_1 q\big)\frac{1}{\mu_2}\Big)\mathbb{1}_{\{R(\sum_{i=2}^{B_1+2}\phi_{1,i},q)>\mu_1 q\}}, \end{aligned} \tag{41}$$

$$\begin{aligned} l_2^{\pi^q}(\phi_{2,1}) =& \Bigg(\phi_{2,1} - q(\mu_1+\mu_2)\Big\lfloor \frac{\sum_{i=2}^{B_1+2}\phi_{1,i}}{q(\mu_1+\mu_2)}\Big\rfloor - R\big(\sum_{i=2}^{B_1+2}\phi_{1,i}, q\big)\frac{\mu_2}{\mu_1}\mathbb{1}_{\{R(\sum_{i=2}^{B_1+2}\phi_{1,i},q)\le\mu_1 q\}} \\ &- \Big(\mu_2 q + \big(R\big(\sum_{i=2}^{B_1+2}\phi_{1,i}, q\big) - \mu_1 q\big)\frac{\mu_1}{\mu_2}\Big)\mathbb{1}_{\{R(\sum_{i=2}^{B_1+2}\phi_{1,i},q)>\mu_1 q\}}\Bigg)\frac{1}{\mu_1}. \end{aligned} \tag{42}$$

Note that $\lim_{q\to 0} R(\phi, q) = 0$ for any finite ϕ, and the expressions (39) and (40) converge to the corresponding expressions in (36) as $q \to 0$. Furthermore, conditions given in (37) and (38) both converge to the conditions given in (35). Thus, $D_1^{\pi^q} \to D_1^{\pi^{mps}}$ as $q \to 0$.

We let s_n^π denote the number of jobs that are processed in the first station but not in the second right after the departure of the n^{th} job from the system under a policy π. If the first station was blocked before the departure of the first job, then we have $\lim_{q\to 0} s_1^{\pi^q} = s_1^{\pi^{mps}} = B_1 + 1$. If the first job leaves before the first station becomes blocked, we have

$$s_1^{\pi^{mps}} = \max\{s : \sum_{i=1}^{s}\phi_{1,i+1} \le \phi_{2,1}\},$$

$$s_1^{\pi^q} = \max\Big\{s : \sum_{i=1}^{s} \phi_{1,i+1} \le q(\mu_1+\mu_2)\Big\lfloor \frac{\phi_{2,1}}{q(\mu_1+\mu_2)}\Big\rfloor + R(\phi_{2,1},q)\frac{\mu_1}{\mu_2}\mathbb{1}_{\{R(\phi_{2,1},q)\le\mu_2 q\}}$$
$$+ (\mu_1 q + (R(\phi_{2,1},q)-\mu_2 q)\frac{\mu_2}{\mu_1})\mathbb{1}_{\{R(\phi_{2,1},q)>\mu_2 q\}}\Big\},$$

and $\lim_{q\to 0} s_1^{\pi^q} = s_1^{\pi^{mps}}$, since $\lim_{q\to 0} R(\phi,q) = 0$.

Similarly, let $\phi^{\pi}_{1,n+s_n^{\pi}+1}$ be the remaining service requirement of the $(n+s_n^{\pi}+1)^{th}$ job at the first station right after the n^{th} job leaves the system under a policy π. If the first station was blocked during the departure of the first job, then we have $\phi^{\pi^q}_{1,B_1+3} = \phi^{\pi^{mps}}_{1,B_1+3} = \phi_{1,B_1+3}$. If the first job leaves before the first station becomes blocked, we have

$$\phi^{\pi^{mps}}_{1,s_1^{\pi^{mps}}+2} = \phi_{1,s_1^{\pi^{mps}}+2} - \Big(\phi_{2,1} - \sum_{i=1}^{s_1^{\pi^{mps}}} \phi_{1,i+1}\Big),$$

$$\phi^{\pi^q}_{1,s_1^{\pi^q}+2} = \phi_{1,s_1^{\pi^q}+2} - \Bigg(q(\mu_1+\mu_2)\Big\lfloor \frac{\phi_{2,1}}{q(\mu_1+\mu_2)}\Big\rfloor + R(\phi_{2,1},q)\frac{\mu_1}{\mu_2}\mathbb{1}_{\{R(\phi_{2,1},q)\le\mu_2 q\}}$$
$$+ (\mu_1 q + (R(\phi_{2,1},q)-\mu_2 q)\frac{\mu_2}{\mu_1})\mathbb{1}_{\{R(\phi_{2,1},q)>\mu_2 q\}} - \sum_{i=1}^{s_1^{\pi^q}} \phi_{1,i+1}\Bigg),$$

and thus $\lim_{q\to 0} \phi^{\pi^q}_{1,s_1+2} = \phi^{\pi^{mps}}_{1,s_1+2}$, since $\lim_{q\to 0} R(\phi,q) = 0$ and $\lim_{q\to 0} s_1^{\pi^q} = s_1^{\pi^{mps}}$.

Assume that $D_n^q \to D_n^{\pi^{mps}}$, $s_n^{\pi^q} \to s_n^{\pi^{mps}}$, and $\phi^{\pi^q}_{1,n+s_n+1} \to \phi^{\pi^{mps}}_{1,n+s_n+1}$ for all $n \le m$ where $m \ge 1$. If $s_m^{\pi} = 0$ under a policy π, then the $(m+1)^{th}$ job is still in the first station when the m^{th} job departs the system and $D^{\pi}_{m+1} = D^{\pi}_m + l^{\pi}(\phi^{\pi}_{1,m+1}) + l^{\pi}(\phi_{2,m+1})$, where $\phi^{\pi}_{1,m+1}$ is the remaining service requirement of the $(m+1)^{th}$ job at the first station under the policy π. Both π^q and π^{mps} assign server 1 to the first station until the $(m+1)^{th}$ job leaves the first station, and we have $\lim_{q\to 0} l^{\pi^q}(\phi^{\pi^q}_{1,m+1}) = l^{\pi^{mps}}(\phi^{\pi^{mps}}_{1,m+1})$ due to our assumption that $\phi^{\pi^q}_{1,m+1} \to \phi^{\pi^{mps}}_{1,m+1}$. Note that once the $(m+1)^{th}$ job finishes being served in the first station, the $(m+2)^{th}$ job enters the system and the first station will be blocked if jobs $n \in \{m+2,\ldots,m+B_1+2\}$ finish being served at the first station before the $(m+1)^{th}$ job departs the system. Thus, expressions similar to (35)–(42) can be written for the $(m+1)^{th}$ job and the jobs

$n \in \{m+2, \ldots, m+B_1+2\}$, and the limit $\lim_{q\to 0} l^{\pi^q}(\phi_{2,m+1}) = l^{\pi^{mps}}(\phi_{2,m+1})$ follows in a similar manner to $\lim_{q\to 0} l^{\pi^q}(\phi_{2,1}) = l^{\pi^{mps}}(\phi_{2,1})$.

Similarly, if $s_m^\pi > 0$, then the $(m+1)^{th}$ job is in the buffer when the m^{th} job departs the system, and we have $D_{m+1}^\pi = D_m^\pi + l^\pi(\phi_{2,m+1})$. Note that at the time the $(m+1)^{th}$ job starts service at the second station, the $(m+s_m^\pi+1)^{th}$ has been partially served at the first station. The first station will be blocked if the jobs $n \in \{m+s_m^\pi+1, \ldots, m+s_m^\pi+B_1+1\}$ finish being served at the first station before the $(m+1)^{th}$ job departs the system. Thus, the limit $\lim_{q\to 0} l^{\pi^q}(\phi_{2,m+1}) = l^{\pi^{mps}}(\phi_{2,m+1})$ follows again in a similar manner, due to our assumptions $s_m^{\pi^q} \to s_m^{\pi^{mps}}$, $\phi_{1,m+s_m^{\pi^q}+1}^{\pi^q} \to \phi_{1,m+s_m^{\pi^{mps}}+1}^{\pi^{mps}}$, and we have

$$\lim_{q\to 0} D_{m+1}^{\pi^{mps}} = D_{m+1}^{\pi^q}.$$

Moreover,

$$s_{m+1}^{\pi^{mps}} = \max\{s : \sum_{i=1}^{s} \phi_{1,i+m+s_m^{\pi^{mps}}}^{\pi^{mps}} \le \phi_{2,m+1}\} + s_m^{\pi^{mps}} - 1,$$

$$\begin{aligned} s_{m+1}^{\pi^q} = \max\Big\{ & s : \sum_{i=1}^{s} \phi_{1,i+m+s_m^{\pi^q}}^{\pi^q} \le q(\mu_1+\mu_2)\Big\lfloor \frac{\phi_{2,m+1}}{q(\mu_1+\mu_2)}\Big\rfloor + R(\phi_{2,m+1},q)\frac{\mu_1}{\mu_2}\mathbb{1}_{\{R(\phi_{2,m+1},q)\le\mu_2 q\}} \\ & + (\mu_1 q + (R(\phi_{2,m+1},q)-\mu_2,q)\frac{\mu_2}{\mu_1})\mathbb{1}_{\{R(\phi_{2,m+1},q)>\mu_2 q\}}\Big\} + s_m^{\pi^q} - 1, \end{aligned}$$

and we have $s_{m+1}^{\pi^q} \to s_{m+1}^{\pi^{mps}}$, due to our assumptions $s_m^{\pi^q} \to s_m^{\pi^{mps}}$, $\phi_{1,m+s_m^{\pi^q}+1}^{\pi^q} \to \phi_{1,m+s_m^{\pi^{mps}}+1}^{\pi^{mps}}$, and the fact that $R(\phi,q) \to 0$ as $q \to 0$. Similarly, $\phi_{1,m+s_{m+1}^{\pi^q}+2}^{\pi^q} \to \phi_{1,m+s_{m+1}^{\pi^{mps}}+2}^{\pi^{mps}}$ and the result follows by induction. □

5.3.2 Performance of Round Robin Policies

Note that Theorem 4 is valid for any buffer size B_1 and implies that performance of the egalitarian processor sharing can be achieved asymptotically for systems of two stations. For larger Markovian networks with homogeneous tasks, we evaluate the performance of the timed round-robin policies through a numerical experiment. We consider timed round-robin (TRR) policies with deterministic (DTRR) and adaptive

(ATRR) time increments. When the time increments are deterministic, a constant parameter q is chosen and $\epsilon = q|V|$ used for the set of busy stations V in step (iii) of the server assignment algorithm in Section 5.3.1, so the average time spent with a server assignment is q for all instances generated. Note that if the network is tandem, the parameter q corresponds to the time between server reassignments as defined in Section 5.3.1. On the other hand, the timed round-robin policy with adaptive time increments uses $q = \frac{1}{\sum_{i=1}^{N} \mu_i}$, where μ_i, $i \in \{1, \ldots, N\}$, are the service rates for a given instance. For each instance, service rates were generated randomly following a U[1,20] distribution.

We consider tree networks with tandem, merge, and split topologies that have equal numbers of servers and stations. For tandem networks, we focus on systems with $N = 3, 4, 5$ stations, since a numerical study of larger systems has impractically high computational requirements. For systems with $N = 3, 4$ stations, 5000 sets of independently generated service rates were used; whereas 200 sets of independently generated service rates were used for systems with $N = 5$ stations. For systems with split and merge topologies, we consider systems with $N = 3$ and $N = 5$ stations, as depicted in Figure 8, and use 1000 and 200 independently generated sets of service rates, respectively. Furthermore, uniform and non-uniform routing probability distributions were considered for all instances with split topologies. We use the probabilities $p_{12} = p_{13} = 0.5$ ($p_{12} = 0.25$, $p_{13} = 0.75$) for uniform (non-uniform) routing in networks with $N = 3$ stations and two-tier networks with $N = 5$ stations. Similarly, we use the probabilities $p_{12} = p_{13} = p_{14} = p_{15} = 0.25$ ($p_{12} = p_{13} = 0.125$, $p_{14} = p_{15} = 0.375$) for uniform (non-uniform) routing in one-tier networks with $N = 5$ stations. For each topology and set of service rates, we use buffer sizes $B_j \in \{1, \ldots, 10\}$ if $N = 3$ and $B_j \in \{1, \ldots, 5\}$ if $N = 4, 5$. When there is a number of stations that are blocked by the same buffer, we assume the station with the lowest index is unblocked when a job moves out of the buffer.

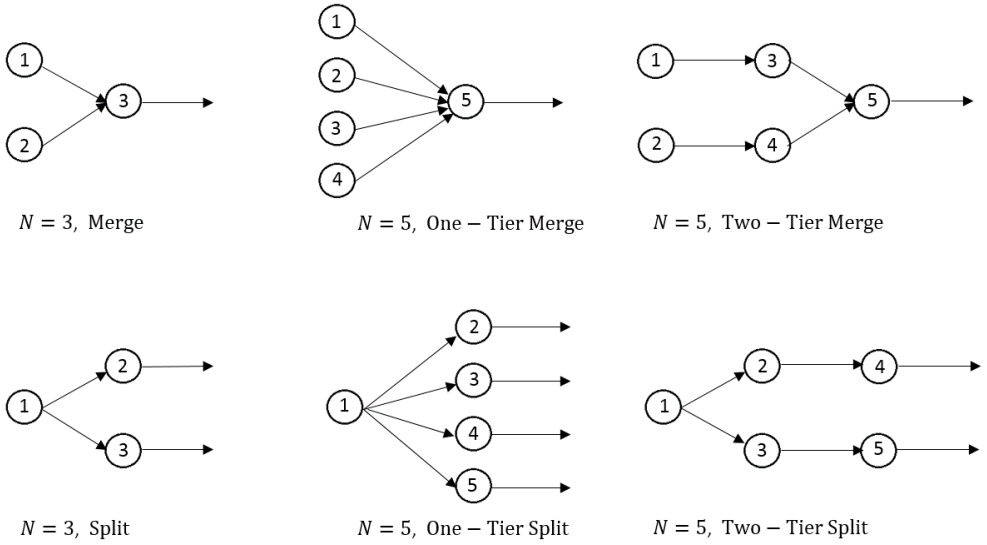

Figure 8: Split and Merge Networks for $N = 3, 5$

The long-run average throughput for the TRR policies were computed through simulation. The simulations were run for TRR policies with deterministic time increments for $q = 0.5, 0.1, 0.05$, as well as adaptive time increments. For tandem networks, we benchmark our results against the long-run average throughputs achieved by the optimal and the best dedicated policies. For each instance, the optimal dynamic server allocation policy is computed using the Policy Iteration algorithm. For non-tandem networks, we use the upper bound yielded by the capacity LP for systems with infinite buffers as our benchmark. For all settings and policies, 95% confidence intervals are reported. Tables 16 through 18 summarize our results for tandem networks, and Tables 19 thorough 24 summarize our results for non-tandem networks.

Our results show that for all system and buffer sizes, the performance of the TRR policies improve as the server reallocations become more frequent. Note that the optimality gap for the TRR policies in tandem networks is not necessarily monotone in buffer size. For $N = 3$, the optimality gap for the TRR policies decreases as buffer size increases from $B_j = 1$ to $B_j = 4$ for $j \in \{1, 2\}$, and increases with buffer size at larger buffer sizes. This behavior is consistent with our observation that

Table 16: Tandem networks with $N = 3$ stations, homogeneous tasks

B	Optimal	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR	Best Ded.
1	8.78±0.06	8.55±0.06	8.62±0.06	8.74±0.06	8.74±0.06	5.01±0.08
2	9.34±0.07	8.89±0.06	9.04±0.06	9.19±0.06	9.20±0.06	5.30±0.09
3	9.69±0.07	9.11±0.07	9.33±0.07	9.48±0.07	9.49±0.07	5.45±0.09
4	9.91±0.08	9.26±0.07	9.55±0.07	9.69±0.07	9.70±0.07	5.55±0.09
5	10.06±0.08	9.38±0.07	9.71±0.07	9.85±0.07	9.85±0.07	5.61±0.10
6	10.16±0.08	9.48±0.08	9.83±0.07	9.96±0.07	9.97±0.08	5.66±0.10
7	10.23±0.08	9.57±0.08	9.93±0.08	10.06±0.08	10.06±0.08	5.69±0.10
8	10.29±0.08	9.64±0.08	10.00±0.08	10.13±0.08	10.14±0.08	5.71±0.10
9	10.33±0.08	9.71±0.08	10.07±0.08	10.19±0.08	10.20±0.08	5.73±0.10
10	10.36±0.08	9.77±0.08	10.12±0.08	10.24±0.08	10.25±0.08	5.74±0.10

Table 17: Tandem networks with $N = 4$ stations, homogeneous tasks

B	Optimal	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR	Best Ded.
1	8.70±0.05	7.36±0.05	7.64±0.05	7.88±0.05	8.05±0.05	4.23±0.07
2	9.35±0.06	7.92±0.06	8.33±0.05	8.63±0.05	8.83±0.06	4.46±0.07
3	9.73±0.06	8.23±0.06	8.76±0.06	9.08±0.06	9.28±0.06	4.58±0.08
4	9.96±0.07	8.43±0.06	9.06±0.06	9.39±0.06	9.59±0.06	4.65±0.08
5	10.11±0.07	8.57±0.06	9.28±0.06	9.61±0.06	9.80±0.07	4.69±0.08

Table 18: Tandem networks with $N = 5$ stations, homogeneous tasks

B	Optimal	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR	Best Ded.
1	8.52±0.20	6.94±0.19	7.21±0.18	7.44±0.19	7.71±0.18	3.61±0.26
2	9.21±0.22	7.45±0.21	7.84±0.20	8.16±0.22	8.52±0.21	3.79±0.29
3	9.61±0.24	7.73±0.22	8.21±0.22	8.60±0.23	9.00±0.22	3.87±0.31
4	9.84±0.25	7.91±0.23	8.48±0.23	8.90±0.24	9.33±0.24	3.91±0.32
5	9.98±0.26	8.03 ±0.24	8.67±0.24	9.12±0.25	9.55±0.24	3.94±0.32

Table 19: Merge networks with $N = 3$ stations, homogeneous tasks

B.	LP Bound	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR
1	14.24±0.23	12.91±0.21	12.92±0.21	13.03±0.21	13.00±0.21
2	14.24±0.23	13.25±0.21	13.28±0.21	13.38±0.21	13.35±0.22
3	14.24±0.23	13.46±0.22	13.50±0.22	13.60±0.22	13.57±0.22
4	14.24±0.23	13.61±0.22	13.65±0.22	13.75±0.22	13.72±0.22
5	14.24±0.23	13.71±0.22	13.76±0.22	13.85±0.22	13.82±0.22
6	14.24±0.23	13.78±0.23	13.84±0.22	13.92±0.22	13.90±0.23
7	14.24±0.23	13.84±0.23	13.89±0.22	13.98±0.23	13.96±0.23
8	14.24±0.23	13.88±0.23	13.94±0.22	14.02±0.23	14.00±0.23
9	14.24±0.23	13.92±0.23	13.97±0.23	14.06±0.23	14.03±0.23
10	14.24±0.23	13.95±0.23	14.00±0.23	14.08±0.23	14.06±0.23

$T^*(0) = T^{\pi^{mps}}(0)$ for systems with $N = 2$ stations and the performance of egalitarian processor sharing as shown in Proposition 14. For larger tandem networks (i.e., for $N = 4, 5$), numerical results show that the optimality gap for the TRR policies decreases as buffer sizes grow. It is expected that the TRR policies perform better for large buffer sizes, since they approximate the egalitarian processor sharing, which yields an upper bound for the infinite buffered systems. However, the optimality gap for a TRR policy increases as the system size increases. In particular, for $q = 0.05$, the optimality gap is about 2% for $B_j = 5, j \in \{1, \ldots, N-1\}$ when $N = 3$, 5%, and 9% for $B_j = 5, j \in \{1, \ldots, N-1\}$, when $N = 4, 5$, respectively.

The adaptive TRR policy performs near-optimal in all tandem settings. In systems with three stations, the largest average optimality gap is 2% across buffer sizes $B_j \in \{1, \ldots, 10\}$. In systems with four and five stations, the optimality gap is 8% and 10% for $B_j = 1, j \in \{1, \ldots, N-1\}$, and reduces to 3% and 4% for $B_j = 5, j \in \{1, \ldots, N-1\}$. Furthermore, the adaptive policy uses the expected first service completion time as the time increment and our results suggest that it is sufficient to change the server assignment after every service completion. Such a server assignment mechanism is advantageous since the server reassignments can be carried out without supervision.

For non-tandem networks, we compare the throughput values simulated for the

Table 20: One-tier merge networks with $N = 5$ stations, homogeneous tasks

B.	LP Bound	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR
1	16.41±0.34	16.19±0.33	16.25±0.33	16.29±0.33	16.25±0.35
2	16.41±0.34	16.28±0.34	16.34±0.34	16.39±0.34	16.34±0.35
3	16.41±0.34	16.33±0.34	16.30±0.34	16.44±0.34	16.39±0.35
4	16.41±0.34	16.36±0.34	16.33±0.34	16.46±0.34	16.41±0.35
5	16.41±0.34	16.37±0.34	16.35±0.34	16.48±0.34	16.43±0.35

Table 21: Two-tier merge networks with $N = 5$ stations, homogeneous tasks

B.	LP Bound	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR
1	15.67±0.35	13.42±0.31	13.57±0.31	13.53±0.31	13.57±0.32
2	15.67±0.35	13.95±0.33	14.18±0.33	14.11±0.33	14.18±0.33
3	15.67±0.35	14.27±0.34	14.35±0.34	14.47±0.34	14.55±0.34
4	15.67±0.35	14.48±0.35	14.58±0.34	14.70±0.34	14.79±0.35
5	15.67±0.35	14.63±0.35	14.75±0.35	14.87±0.35	14.96±0.35

TRR policies to the upper bound computed for the systems with infinite buffers. Confidence intervals on the capacity upper bound are computed over randomly generated instances. As expected, the long-run average throughout achieved by the TRR policies approaches the processor sharing upper bound as the time increment q decreases and the buffer sizes increase. For one-tier networks with merge topology, the performance of the TRR policies relative to the upper bound increases with system size. In particular, the TRR policy with $q = 0.05$ on average achieves 92% of the upper bound for merge networks with $N = 3$, $B_j = 1$, $j \in \{1, 2\}$ whereas it achieves 99% for one-tier merge networks with $N = 5$, $B_j = 1$, $j \in \{1, \ldots, 4\}$. In two-tier merge networks with $N = 5$ the round-robin policies yield comparatively lower throughput than the capacity upper bound. Furthermore, the performance of the TRR policies show a greater dependency on the buffer sizes, yielding 81% and 95% of the the capacity upper bound for buffer sizes $B_j = 1$ and $B_j = 5$, $j \in \{1, \ldots, 4\}$, respectively. We also observe that the results in two-tier merge networks are comparable to a tandem line with $N = 3$ stations. Note that Proposition 11 implies that in a tandem line with homogeneous tasks and equal number of servers and stations, the capacity upper bound is equal to the average service rate. Thus, for the tandem lines we consider for Tables 16–18, the expected upper bound is 10.5, and the TRR policy

Table 22: Split networks with $N = 3$ stations, homogeneous tasks

Equal split at the fork					
B	LP Bound	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR
1	14.24±0.23	11.93±0.19	11.92±0.19	12.00±0.19	11.94±0.19
2	14.24±0.23	12.61±0.20	12.62±0.20	12.71±0.20	12.67±0.21
3	14.24±0.23	13.00±0.21	13.05±0.21	13.13±0.21	13.09±0.21
4	14.24±0.23	13.25±0.22	13.31±0.22	13.40±0.22	13.37±0.22
5	14.24±0.23	13.40±0.22	13.50±0.22	13.58±0.22	13.56±0.22
6	14.24±0.23	13.52±0.22	13.63±0.22	13.71±0.22	13.69±0.22
7	14.24±0.23	13.60±0.22	13.72±0.22	13.80±0.22	13.79±0.23
8	14.24±0.23	13.67±0.23	13.80±0.22	13.87±0.23	13.86±0.23
9	14.24±0.23	13.72±0.23	13.85±0.23	13.93±0.23	13.92±0.23
10	14.24±0.23	13.76±0.23	13.89±0.23	13.97±0.23	13.96±0.23
Unequal split at the fork					
B	LP Bound	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR
1	14.04±0.23	11.50±0.19	11.52±0.19	11.62±0.19	11.57±0.19
2	14.04±0.23	12.13±0.20	12.18±0.20	12.27±0.20	12.24±0.21
3	14.04±0.23	12.57±0.21	12.58±0.21	12.67±0.21	12.65±0.21
4	14.04±0.23	12.76±0.22	12.85±0.22	12.95±0.22	12.92±0.22
5	14.04±0.23	12.94±0.22	13.04±0.22	13.12±0.22	13.12±0.23
6	14.04±0.23	13.08±0.23	13.18±0.22	13.27±0.23	13.26±0.23
7	14.04±0.23	13.18±0.23	13.30±0.23	13.37±0.23	13.37±0.23
8	14.04±0.23	13.26±0.23	13.38±0.23	13.46±0.23	13.46±0.23
9	14.04±0.23	13.33±0.23	13.45±0.23	13.53±0.23	13.53±0.23
10	14.04±0.23	13.38±0.23	13.51±0.23	13.59±0.23	13.59±0.23

with $q = 0.05$ achieves 83% and 94% of the upper bound for buffer sizes $B_j = 1$ and $B_j = 5$, $j \in \{1, 2\}$, respectively. Once again, the adaptive TRR policy performs near-optimal in all settings, as it on average yields 96%, 99%, and 92% of the capacity upper bound for merge networks with $N = 3$, one-tier merge networks with $N = 5$, and two-tier merge networks with $N = 5$, respectively.

Our results suggest that the performances of the TRR policies in networks with merge and split topologies are similar for large buffer sizes. On the other hand, throughput achieved split networks is more sensitive to buffer size and the performance of the TRR policy with $q = 0.05$ is lower in split networks for buffer size $B_j = 1, j \in \{1, \ldots, N-1\}$. In particular, it yields 84%, 92%, and 79% of the capacity upper bound when the routing probabilities are uniform for split networks with $N = 3$, one-tier split networks with $N = 5$, and two-tier split networks with $N = 5$, respectively. The performance gap observed for merge and split topologies further grows when the jobs is routed non-uniformly in a split network. On average, the TRR

Table 23: One-tier split networks with $N = 5$ stations, homogeneous tasks

Equal split at the fork					
B	LP Bound	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR
1	16.41±0.34	15.10±0.32	15.16±0.32	15.19±0.32	15.15±0.33
2	16.41±0.34	15.86±0.33	15.91±0.33	15.94±0.33	15.91±0.34
3	16.41±0.34	16.14±0.33	16.12±0.33	16.24±0.33	16.21±0.35
4	16.41±0.34	16.27±0.34	16.25±0.34	16.37±0.34	16.34±0.35
5	16.41±0.34	16.33±0.34	16.31±0.34	16.43±0.34	16.40±0.35
Unequal split at the fork					
B	LP Bound	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR
1	16.39±0.33	14.66±0.31	14.70±0.31	14.74±0.31	14.70±0.32
2	16.39±0.33	15.46±0.32	15.52±0.32	15.55±0.32	15.52±0.34
3	16.39±0.33	15.84±0.33	15.83±0.33	15.95±0.33	15.91±0.34
4	16.39±0.33	16.04±0.33	16.03±0.33	16.15±0.33	16.12±0.35
5	16.39±0.33	16.16±0.33	16.15±0.33	16.26±0.33	16.23±0.35

Table 24: Two-tier split networks with $N = 5$ stations, homogeneous tasks

Equal split at the fork					
B	LP Bound	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR
1	15.67±0.35	12.39±0.29	12.39±0.29	12.43±0.29	12.39±0.30
2	15.67±0.35	13.38±0.33	13.42±0.32	13.43±0.32	13.42±0.32
3	15.67±0.35	13.94±0.34	13.96±0.34	14.02±0.34	14.02±0.34
4	15.67±0.35	14.28±0.36	14.32±0.35	14.39±0.35	14.41±0.35
5	15.67±0.35	14.49±0.36	14.56±0.36	14.64±0.36	14.68±0.36
Unequal split at the fork					
B	LP Bound	DTRR($q = 0.5$)	DTRR($q = 0.1$)	DTRR($q = 0.05$)	ATRR
1	15.39±0.36	11.69±0.27	11.71±0.27	11.74±0.27	11.71±0.27
2	15.39±0.36	12.61±0.30	12.66±0.30	12.67±0.30	12.66±0.30
3	15.39±0.36	13.17±0.32	13.19±0.31	13.25±0.31	13.24±0.32
4	15.39±0.36	13.54±0.33	13.58±0.33	13.64±0.33	13.64±0.33
5	15.39±0.36	13.80±0.34	13.85±0.34	13.92±0.34	13.93±0.34

policy with $q = 0.05$ yields 2%–3% lower performance with respect to the capacity upper bound when the routing probabilities are non-uniform.

As with the tandem and merge topologies, the adaptive TRR policy results in near-optimal long-run average throughputs in networks with split topology. Overall, ATRR policies achieve 88%–98% in split networks with uniform routing probabilities and 85%–96% in split networks with non-uniform routing topologies. Therefore, our experiments suggest that the ATRR policy can be utilizes as an easy-to-implement server assignment heuristic in queueing networks with various topologies.

5.4 Conclusions

We study queueing networks with general routing and flexible servers. We introduce a novel processor sharing model for queueing networks where servers are shared by multiple nodes in the network. For systems with infinite buffers and non-collaborative flexible servers, we develop a linear program that yields an upper bound on the attainable long-run average throughput where the solution to the LP can be interpreted to correspond to a processor sharing policy. For tandem lines with homogeneous tasks and infinite buffers, we identify the optimal processor sharing policy. For systems with two stations and non-homogeneous tasks, we prove that either a processor sharing policy or a dedicated server assignment is optimal. Furthermore, we show that the maximum throughput the linear program yields for non-collaborative systems can be improved by allowing servers to collaborate only if the tasks are non-homogeneous.

We analyze the performance of the processor sharing policies in networks with finite buffers. For Markovian non-collaborative systems with two stations, two servers, and a finite buffer between the stations, we prove that the long-run average throughput attained by a processor sharing policy converges to the optimal throughput as buffer size grows. For non-collaborative systems where processor sharing is not implementable, we develop timed round-robin server assignment policies. We show that

these policies approximate processor sharing as the server reassignments become more frequent in systems with two stations and finite buffers. For larger queueing networks with either tandem, merge or split topologies, our numerical results suggest that the performance of the round-robin policies improves with frequent server reassignments. In particular, we propose an adaptive round-robin policy that achieves near-optimal throughputs in all topologies. Furthermore, the throughput attained by the timed round-robin policies approaches to the upper-bound yielded by the capacity LP as the buffer sizes grow. We conclude that flexible servers can be utilized in a simple round-robin fashion to achieve near-optimal performance in non-collaborative systems when the optimal policy is difficult to identify.

CHAPTER VI

CONCLUSION

This dissertation provides an analysis on the benefits of cross-training in non-collaborative manufacturing systems. In particular, we demonstrate that cross-training can improve performance measures through dynamic allocation of workers to tasks in systems where multiple servers cannot be accommodated at a station. We focus on three different models and study dynamic server allocation (i) in a tandem line to improve throughput, (ii) in systems where frequent reallocation of the servers is undesirable, and (iii) in queueing networks with general routing.

In Chapter 3, we study a non-collaborative tandem queueing network with flexible servers. A comparison of collaborative and non-collaborative tandem lines shows that the importance of collaboration is dependent on the specialization levels of the servers. If the performance of the servers is not dependent on the task, dynamic server allocation can compensate for the lack of collaboration in systems with large buffers. We also prove that a threshold-type server assignment policy is throughput-optimal for small tandem lines with two stations, and identify the conditions for the optimal policy to be unique. Our numerical results suggest that the optimal policy maintains a similar threshold structure in systems with more than two stations. Moreover, heuristic policies that locally reassign the servers to balance the workload perform near-optimal, whereas the performance of dedicated server assignment policies remains poor. Thus, we conclude that server flexibility is beneficial even when collaboration is not an option.

We also consider systems in which frequent reassignment of the servers is not desirable due to setups. Our work presented in Chapter 4 shows that the extent to

which the server flexibility must be utilized depends on the setup costs. For systems with two stations and constant setup costs, we show that a double-threshold policy maximizes the long-run average profit. Furthermore, dynamic server allocation still yields significant improvement over dedicated server assignment policies for moderate setup costs. Our results indicate that if the setup costs are large, the optimal policy becomes dedicated and avoids setups. However, disadvantages of frequent server reassignments can be countered by increasing the capacity for the work-in-process inventory, and cross-training becomes beneficial for larger buffer sizes. In the presence of setup costs, the performance of dynamic server assignment policies is also dependent on the relative difficulties of the tasks. In particular, cross-training is most beneficial when the tasks are of similar difficulty and service rates do not depend on the task.

Finally, we address the allocation of flexible servers in queueing networks with general topology and routing in Chapter 5. We propose a new processor sharing model for queueing networks with flexible servers, where the optimal processor sharing policy achieves the maximal throughput in systems with infinite buffers. For non-collaborative networks where servers cannot be shared, we develop a class of round-robin server assignment policies. Moreover, we show that the round-robin policies converge to the processor sharing scheme for tandem networks with two stations and homogeneous tasks. We also evaluate the performance of these round-robin policies via numerical experiments in finite-buffered networks with tandem, merge, or split topologies. Our computational results suggest that the round-robin policies are near-optimal. Thus, processor sharing can be used to develop simple server assignment heuristics for general non-collaborative networks with finite buffers where the optimal policy has a complex structure.

CHAPTER VII

FUTURE RESEARCH

Our findings so far show that cross-training is valuable in various non-collaborative systems. Further insights can be gained through the study of different flexibility or collaboration structures in manufacturing systems. Here we present three possible extensions.

Analysis of tandem manufacturing systems in this dissertation allows for a fully staffed production line where each station can be assigned a server. However, obtaining analytical results for systems with more than two stations is challenging due to the high dimensionality of the server allocation problem. Furthermore, understaffed manufacturing lines are commonplace in industries where labor costs and number of tasks in the system are both high. Thus, it is of interest to characterize how flexible servers should be utilized in larger tandem networks that are understaffed. There is only limited literature on how the dynamic allocation of cross-trained servers should be done in collaborative understaffed systems. However, our experience with the non-collaborative setting leads us to believe that the structure of the optimal policies might be easier to characterize for non-collaborative systems. Thus, this is a promising research direction.

Another extension of our current work is to consider systems with partially flexible servers. In many manufacturing and service systems, full flexibility of the servers is not possible due to a limited cross-training budget or limited mobility of the servers between workstations. Therefore, in these systems one or more servers are confined to work at a single station or at a group of stations. Similar flexibility structures can

be used in assembly/disassembly systems where the dedicated servers handle assembly or disassembly operations while intermediate stations can be served by flexible servers. We expect that for partially flexible systems, optimal dynamic server allocation policies will still follow a threshold structure. Furthermore, partially dedicated systems result in smaller action spaces than their fully flexible counterparts. In the light of these observations, we hope to obtain analytical results for partially flexible systems with more than two stations.

The final future research direction we propose is to examine manufacturing systems where collaboration is allowed only at a subset of stations. Our motivation to study non-collaborative queueing networks is to design server allocation policies for manufacturing systems with insufficient tooling or other physical limitations. These restrictions often do not affect all operations in a manufacturing system, especially when the operations are significantly different. For instance, multiple servers can simultaneously utilize workstations designated for packaging or inspection operations while earlier stages of production may suffer from insufficient tooling. Thus, analysis of systems with a mixture of collaborative and non-collaborative stations is a potential extension to our current work.

APPENDIX A

SYSTEMS WITH NON-HOMOGENEOUS TASKS AND CONSTANT SETUP COSTS

Table 25: Systems with non-homogeneous tasks and a dominating server, setup cost $2c = 1$

B	Optimal	BD	HR	HST
1	8.42 ± 0.17	7.44 ± 0.18	8.12 ± 0.16	8.12 ± 0.16
2	8.87 ± 0.18	7.81 ± 0.20	8.68 ± 0.17	8.68 ± 0.17
3	9.15 ± 0.19	8.04 ± 0.20	8.96 ± 0.18	8.96 ± 0.18
4	9.34 ± 0.20	8.19 ± 0.21	9.14 ± 0.19	9.14 ± 0.19
5	9.46 ± 0.20	8.29 ± 0.22	9.29 ± 0.19	9.29 ± 0.19
6	9.55 ± 0.20	8.37 ± 0.22	9.39 ± 0.20	9.39 ± 0.20
7	9.62 ± 0.21	8.42 ± 0.22	9.47 ± 0.20	9.48 ± 0.20
8	9.68 ± 0.21	8.46 ± 0.22	9.54 ± 0.20	9.54 ± 0.20
9	9.72 ± 0.21	8.49 ± 0.23	9.59 ± 0.20	9.59 ± 0.20
10	9.75 ± 0.21	8.52 ± 0.23	9.64 ± 0.21	9.64 ± 0.21
Aggregate	9.36 ± 0.06	8.20 ± 0.06	9.18 ± 0.06	9.18 ± 0.06
B	P-t-B(all)	P-t-B(HR-BD)	P-t-B(HST-BD)	P-t-B(HR-HST)
1	8.29 ± 0.17	8.29 ± 0.17	8.29 ± 0.17	8.12 ± 0.16
2	8.80 ± 0.18	8.80 ± 0.18	8.80 ± 0.18	8.68 ± 0.17
3	9.08 ± 0.19	9.08 ± 0.19	9.08 ± 0.19	8.96 ± 0.18
4	9.25 ± 0.19	9.25 ± 0.19	9.25 ± 0.19	9.14 ± 0.19
5	9.39 ± 0.20	9.39 ± 0.20	9.39 ± 0.20	9.29 ± 0.19
6	9.49 ± 0.20	9.48 ± 0.20	9.48 ± 0.20	9.39 ± 0.20
7	9.56 ± 0.20	9.56 ± 0.20	9.56 ± 0.20	9.48 ± 0.20
8	9.62 ± 0.21	9.62 ± 0.21	9.62 ± 0.21	9.54 ± 0.20
9	9.66 ± 0.21	9.66 ± 0.21	9.66 ± 0.21	9.59 ± 0.20
10	9.70 ± 0.21	9.70 ± 0.21	9.70 ± 0.21	9.64 ± 0.21
Aggregate	9.28 ± 0.06	9.28 ± 0.06	9.28 ± 0.06	9.18 ± 0.06

Table 26: Systems with non-homogeneous tasks and a dominating server, setup cost $2c = 5$

B	Optimal	BD	HR	HST
1	7.89 ± 0.17	7.44 ± 0.18	7.34 ± 0.16	7.33 ± 0.16
2	8.46 ± 0.18	7.81 ± 0.20	8.03 ± 0.17	8.04 ± 0.17
3	8.82 ± 0.19	8.04 ± 0.20	8.54 ± 0.18	8.54 ± 0.18
4	9.06 ± 0.19	8.19 ± 0.21	8.85 ± 0.19	8.85 ± 0.19
5	9.23 ± 0.20	8.29 ± 0.22	9.03 ± 0.19	9.03 ± 0.19
6	9.35 ± 0.20	8.37 ± 0.22	9.17 ± 0.19	9.18 ± 0.19
7	9.45 ± 0.20	8.42 ± 0.22	9.29 ± 0.20	9.29 ± 0.20
8	9.52 ± 0.21	8.46 ± 0.22	9.37 ± 0.20	9.37 ± 0.20
9	9.58 ± 0.21	8.49 ± 0.23	9.44 ± 0.20	9.44 ± 0.20
10	9.62 ± 0.21	8.52 ± 0.23	9.50 ± 0.20	9.50 ± 0.20
Aggregate	9.10 ± 0.06	8.20 ± 0.07	8.85 ± 0.06	8.86 ± 0.06
B	P-t-B(all)	P-t-B(HR-BD)	P-t-B(HST-BD)	P-t-B(HR-HST)
1	7.87 ± 0.17	7.87 ± 0.17	7.87 ± 0.17	7.34 ± 0.16
2	8.40 ± 0.18	8.40 ± 0.18	8.40 ± 0.18	8.05 ± 0.17
3	8.78 ± 0.19	8.78 ± 0.19	8.78 ± 0.19	8.54 ± 0.18
4	9.03 ± 0.19	9.03 ± 0.19	9.03 ± 0.19	± 8.85 0.19
5	9.19 ± 0.20	9.19 ± 0.20	9.19 ± 0.20	9.04 ± 0.19
6	9.31 ± 0.20	9.31 ± 0.20	9.31 ± 0.20	9.18 ± 0.19
7	9.41 ± 0.20	9.40 ± 0.20	9.40 ± 0.20	9.29 ± 0.20
8	9.48 ± 0.21	9.48 ± 0.21	9.48 ± 0.21	9.38 ± 0.20
9	9.54 ± 0.21	9.54 ± 0.21	9.54 ± 0.21	9.45 ± 0.20
10	9.59 ± 0.21	9.59 ± 0.21	9.59 ± 0.21	9.51 ± 0.20
Aggregate	9.06 ± 0.06	9.06 ± 0.06	9.06 ± 0.06	8.86 ± 0.06

Table 27: Systems with non-homogeneous tasks and a dominating server, setup cost $2c = 20$

B	Optimal	BD	HR	HST
1	7.44 ± 0.18	7.44 ± 0.18	5.04 ± 0.18	4.92 ± 0.17
2	7.87 ± 0.19	7.81 ± 0.20	6.21 ± 0.17	6.18 ± 0.17
3	8.19 ± 0.20	8.04 ± 0.20	7.04 ± 0.18	7.04 ± 0.17
4	8.48 ± 0.20	8.19 ± 0.21	7.65 ± 0.18	7.65 ± 0.18
5	8.68 ± 0.20	8.29 ± 0.22	8.13 ± 0.19	8.11 ± 0.19
6	8.86 ± 0.21	8.37 ± 0.22	8.47 ± 0.20	8.45 ± 0.20
7	9.00 ± 0.21	8.42 ± 0.22	8.70 ± 0.20	8.68 ± 0.20
8	9.11 ± 0.21	8.46 ± 0.22	8.86 ± 0.20	8.85 ± 0.20
9	9.20 ± 0.21	8.49 ± 0.23	8.99 ± 0.20	8.98 ± 0.20
10	9.28 ± 0.21	8.52 ± 0.23	9.09 ± 0.20	9.09 ± 0.20
Aggregate	8.61 ± 0.06	8.20 ± 0.07	7.82 ± 0.07	7.80 ± 0.07
B	P-t-B(all)	P-t-B(HR-BD)	P-t-B(HST-BD)	P-t-B(HR-HST)
1	7.44 ± 0.18	7.44 ± 0.18	7.44 ± 0.18	5.13 ± 0.18
2	7.87 ± 0.19	7.87 ± 0.19	7.87 ± 0.19	6.22 ± 0.17
3	8.18 ± 0.20	8.18 ± 0.20	8.18 ± 0.20	7.05 ± 0.17
4	8.43 ± 0.20	8.43 ± 0.20	8.43 ± 0.20	7.65 ± 0.18
5	8.65 ± 0.20	8.64 ± 0.20	8.64 ± 0.20	8.13 ± 0.19
6	8.83 ± 0.21	8.83 ± 0.21	8.82 ± 0.21	8.47 ± 0.20
7	8.98 ± 0.21	8.97 ± 0.21	8.97 ± 0.21	8.70 ± 0.20
8	9.09 ± 0.21	9.09 ± 0.21	9.08 ± 0.21	8.87 ± 0.20
9	9.18 ± 0.21	9.18 ± 0.21	9.17 ± 0.21	8.99 ± 0.20
10	9.26 ± 0.21	9.26 ± 0.21	9.25 ± 0.21	9.10 ± 0.20
Aggregate	8.59 ± 0.07	8.58 ± 0.07	8.58 ± 0.07	7.83 ± 0.07

APPENDIX B

SYSTEMS WITH NON-CONSTANT SETUP COSTS

Table 28: Systems with homogeneous tasks non-constant setup costs

B	Optimal	BD	Avg. C.	S. of C.	Max. C.	Min. C.	Two-T.
1	7.20 ± 0.04	6.35 ± 0.05	6.57 ± 0.05	6.72 ± 0.05	6.61 ± 0.05	6.47 ± 0.05	6.59 ± 0.05
2	7.94 ± 0.05	6.65 ± 0.05	7.18 ± 0.05	7.24 ± 0.05	7.20 ± 0.05	7.14 ± 0.05	7.18 ± 0.05
3	8.50 ± 0.05	6.83 ± 0.05	7.59 ± 0.05	7.60 ± 0.05	7.59 ± 0.05	7.59 ± 0.05	7.58 ± 0.05
4	8.90 ± 0.05	6.95 ± 0.06	7.89 ± 0.05	7.85 ± 0.05	7.87 ± 0.05	7.87 ± 0.05	7.86 ± 0.05
5	9.19 ± 0.05	7.03 ± 0.06	8.09 ± 0.06	8.06 ± 0.06	8.08 ± 0.06	8.07 ± 0.06	8.07 ± 0.06
6	9.40 ± 0.05	7.08 ± 0.06	8.23 ± 0.06	8.21 ± 0.06	8.23 ± 0.06	8.21 ± 0.06	8.22 ± 0.06
7	9.56 ± 0.05	7.13 ± 0.06	8.34 ± 0.06	8.32 ± 0.06	8.33 ± 0.06	8.32 ± 0.06	8.33 ± 0.06
8	9.68 ± 0.05	7.16 ± 0.06	8.42 ± 0.06	8.41 ± 0.06	8.42 ± 0.06	8.40 ± 0.06	8.41 ± 0.06
9	9.78 ± 0.05	7.18 ± 0.06	8.48 ± 0.06	8.48 ± 0.06	8.48 ± 0.06	8.47 ± 0.06	8.47 ± 0.06
10	9.86 ± 0.05	7.20 ± 0.06	8.53 ± 0.06	8.53 ± 0.06	8.53 ± 0.06	8.52 ± 0.06	8.52 ± 0.06
Agg.	9.00 ± 0.02	6.96 ± 0.02	7.93 ± 0.02	7.94 ± 0.02	7.94 ± 0.02	7.91 ± 0.02	7.92 ± 0.02

Table 29: Systems with non-homogeneous tasks and a dominating server, non-constant setup costs

B	Optimal	BD	HR-Avg. C.	HR-S. of C.	HR-Max. C.	HR-Min. C.
1	7.63 ± 0.04	7.44 ± 0.04	6.33 ± 0.04	7.44 ± 0.04	6.44 ± 0.04	6.15 ± 0.04
2	8.16 ± 0.04	7.81 ± 0.04	7.32 ± 0.04	7.81 ± 0.04	7.34 ± 0.0.4	7.25 ± 0.04
3	8.53 ± 0.04	8.04 ± 0.05	7.95 ± 0.04	8.16 ± 0.04	7.95 ± 0.0.4	7.95 ± 0.04
4	8.80 ± 0.04	8.19 ± 0.05	8.41 ± 0.04	8.59 ± 0.04	8.39 ± 0.04	8.37 ± 0.04
5	9.00 ± 0.04	8.29 ± 0.05	8.71 ± 0.04	8.74 ± 0.04	8.70 ± 0.04	8.68 ± 0.04
6	9.15 ± 0.05	8.37 ± 0.05	8.92 ± 0.04	8.86 ± 0.04	8.92 ± 0.04	8.87 ± 0.04
7	9.26 ± 0.05	8.42 ± 0.05	9.06 ± 0.04	8.98 ± 0.04	9.07 ± 0.04	9.02 ± 0.04
8	9.35 ± 0.05	8.46 ± 0.05	9.18 ± 0.04	9.08 ± 0.04	9.18 ± 0.04	9.15 ± 0.04
9	9.43 ± 0.05	8.49 ± 0.05	9.27 ± 0.05	9.17 ± 0.04	9.27 ± 0.05	9.24 ± 0.04
10	9.48 ± 0.05	8.52 ± 0.05	9.34 ± 0.05	9.25 ± 0.05	9.35 ± 0.05	9.32 ± 0.05
Agg.	8.88 ± 0.01	8.20 ± 0.02	8.45 ± 0.01	8.61 ± 0.01	8.46 ± 0.01	8.40 ± 0.01
B	HR-Two-T.	HST-Avg. C.	HST-S. of C.	HST-Max. C.	HST-Min. C.	HST-Two-T.
1	6.31 ± 0.04	6.66 ± 0.04	6.40 ± 0.04	6.16 ± 0.04	6.30 ± 0.04	6.30 ± 0.04
2	7.32 ± 0.04	7.41 ± 0.04	7.34 ± 0.04	7.26 ± 0.04	7.26 ± 0.04	7.26 ± 0.04
3	7.95 ± 0.04	7.96 ± 0.04	7.95 ± 0.04	7.95 ± 0.04	7.89 ± 0.04	7.89 ± 0.04
4	8.40 ± 0.04	8.36 ± 0.04	8.38 ± 0.04	8.38 ± 0.04	8.34 ± 0.04	8.34 ± 0.04
5	8.71 ± 0.04	8.65 ± 0.04	8.69 ± 0.04	8.68 ± 0.04	8.67 ± 0.04	8.67 ± 0.04
6	8.92 ± 0.04	8.88 ± 0.04	8.91 ± 0.04	8.88 ± 0.04	8.89 ± 0.04	8.89 ± 0.04
7	9.06 ± 0.04	9.04 ± 0.04	9.06 ± 0.04	9.04 ± 0.04	9.04 ± 0.04	9.04 ± 0.04
8	9.18 ± 0.04	9.17 ± 0.04	9.18 ± 0.04	9.16 ± 0.04	9.17 ± 0.04	9.16 ± 0.04
9	9.27 ± 0.05	9.26 ± 0.05	9.27 ± 0.05	9.25 ± 0.05	9.26 ± 0.05	9.26 ± 0.05
10	9.35 ± 0.05	9.34 ± 0.05	9.35 ± 0.05	9.33 ± 0.05	9.34 ± 0.05	9.34 ± 0.05
Agg.	8.45 ± 0.01	8.47 ± 0.01	8.45 ± 0.01	8.41 ± 0.01	8.42 ± 0.01	8.41 ± 0.01

Table 30: Systems with homogeneous tasks non-constant setup costs - (P-t-B)

B	Optimal	BD	Avg. C.	S. of C.	Max. C.	Min. C.	Two-T.
1	7.20 ± 0.04	6.35 ± 0.05	6.76 ± 0.05	6.76 ± 0.05	6.76 ± 0.05	6.74 ± 0.05	6.75 ± 0.05
2	7.94 ± 0.05	6.65 ± 0.05	7.27 ± 0.05	7.27 ± 0.05	7.27 ± 0.05	7.26 ± 0.05	7.25 ± 0.05
3	8.50 ± 0.05	6.83 ± 0.05	7.64 ± 0.05	7.62 ± 0.05	7.63 ± 0.05	7.64 ± 0.05	7.61 ± 0.05
4	8.90 ± 0.05	6.95 ± 0.05	7.91 ± 0.05	7.87 ± 0.05	7.90 ± 0.05	7.90 ± 0.05	7.88 ± 0.05
5	9.19 ± 0.05	7.03 ± 0.06	8.10 ± 0.06	8.07 ± 0.06	8.09 ± 0.06	8.09 ± 0.06	8.08 ± 0.06
6	9.40 ± 0.05	7.08 ± 0.06	8.24 ± 0.06	8.21 ± 0.06	8.24 ± 0.06	8.23 ± 0.06	8.23 ± 0.06
7	9.56 ± 0.05	7.13 ± 0.06	8.34 ± 0.06	8.33 ± 0.06	8.34 ± 0.06	8.32 ± 0.06	8.33 ± 0.06
8	9.68 ± 0.05	7.16 ± 0.06	8.42 ± 0.06	8.41 ± 0.06	8.42 ± 0.06	8.41 ± 0.06	8.41 ± 0.06
9	9.78 ± 0.05	7.18 ± 0.06	8.48 ± 0.06	8.48 ± 0.06	8.48 ± 0.06	8.47 ± 0.06	8.48 ± 0.06
10	9.86 ± 0.05	7.20 ± 0.06	8.53 ± 0.06	8.53 ± 0.06	8.54 ± 0.06	8.52 ± 0.06	8.53 ± 0.06
Agg.	9.00 ± 0.02	6.96 ± 0.02	7.97 ± 0.02	7.95 ± 0.02	7.97 ± 0.02	7.96 ± 0.02	7.96 ± 0.02

Table 31: Systems with non-homogeneous tasks and a dominating server, non-constant setup costs - (P-t-B)

B	Optimal	BD	HR-Avg. C.	HR-S. of C.	HR-Max. C.	HR-Min. C.
1	7.63 ± 0.04	7.44 ± 0.04	7.62 ± 0.04	7.44 ± 0.04	7.62 ± 0.04	7.60 ± 0.04
2	8.16 ± 0.04	7.81 ± 0.04	8.13 ± 0.04	7.81 ± 0.04	8.13 ± 0.0.4	8.11 ± 0.04
3	8.53 ± 0.04	8.04 ± 0.05	8.49 ± 0.04	8.16 ± 0.04	8.48 ± 0.0.4	8.49 ± 0.04
4	8.80 ± 0.04	8.19 ± 0.05	8.76 ± 0.04	8.62 ± 0.04	8.75 ± 0.04	8.75 ± 0.04
5	9.00 ± 0.04	8.29 ± 0.05	8.97 ± 0.04	8.86 ± 0.04	8.96 ± 0.04	8.95 ± 0.04
6	9.15 ± 0.05	8.37 ± 0.05	9.12 ± 0.05	9.02 ± 0.04	9.12 ± 0.05	9.10 ± 0.05
7	9.26 ± 0.05	8.42 ± 0.05	9.23 ± 0.05	9.14 ± 0.05	9.23 ± 0.05	9.22 ± 0.05
8	9.35 ± 0.05	8.46 ± 0.05	9.32 ± 0.05	9.23 ± 0.05	9.32 ± 0.05	9.31 ± 0.05
9	9.43 ± 0.05	8.49 ± 0.05	9.40 ± 0.05	9.31 ± 0.04	9.40 ± 0.05	9.38 ± 0.04
10	9.48 ± 0.05	8.52 ± 0.05	9.46 ± 0.05	9.37 ± 0.05	9.46 ± 0.05	9.44 ± 0.05
Agg.	8.88 ± 0.01	8.20 ± 0.02	8.85 ± 0.01	8.70 ± 0.01	8.85 ± 0.01	8.83 ± 0.01
B	HR-Two-T.	HST-Avg. C.	HST-S. of C.	HST-Max. C.	HST-Min. C.	HST-Two-T.
1	7.62 ± 0.04	7.62 ± 0.04	7.62 ± 0.04	7.61 ± 0.04	7.61 ± 0.04	7.61 ± 0.04
2	8.13 ± 0.04	8.13 ± 0.04	8.13 ± 0.04	8.11 ± 0.04	8.12 ± 0.04	8.12 ± 0.04
3	8.49 ± 0.04	8.48 ± 0.04	8.48 ± 0.04	8.49 ± 0.04	8.47 ± 0.04	8.47 ± 0.04
4	8.76 ± 0.04	8.73 ± 0.04	8.75 ± 0.04	8.75 ± 0.04	8.74 ± 0.04	8.74 ± 0.04
5	8.97 ± 0.04	8.93 ± 0.04	8.96 ± 0.04	8.95 ± 0.04	8.95 ± 0.04	8.95 ± 0.04
6	9.12 ± 0.05	9.09 ± 0.05	9.11 ± 0.05	9.10 ± 0.05	9.10 ± 0.05	9.11 ± 0.05
7	9.23 ± 0.05	9.21 ± 0.05	9.23 ± 0.05	9.22 ± 0.05	9.22 ± 0.05	9.22 ± 0.05
8	9.32 ± 0.04	9.31 ± 0.05	9.32 ± 0.05	9.31 ± 0.05	9.31 ± 0.05	9.31 ± 0.05
9	9.40 ± 0.05	9.38 ± 0.05	9.39 ± 0.05	9.38 ± 0.05	9.39 ± 0.05	9.39 ± 0.05
10	9.46 ± 0.05	9.45 ± 0.05	9.45 ± 0.05	9.44 ± 0.05	9.45 ± 0.05	9.45 ± 0.05
Agg.	8.85 ± 0.01	8.83 ± 0.01	8.85 ± 0.01	8.84 ± 0.01	8.84 ± 0.01	8.84 ± 0.01

REFERENCES

[1] Agnihothri, S. R., Mishra, A. K., and Simmons, D. E., "Workforce cross-training decisions in field service systems with two job types," *Journal of the Operational Research Society*, vol. 54, pp. 410–418, Apr. 2003.

[2] Ahn, H.-S., Duenyas, I., and Lewis, M., "Two-stage tandem queuing system with flexible servers," *Probability in the Engineering and Informational Sciences*, vol. 16, pp. 453–469, 2002.

[3] Ahn, H.-S., Duenyas, I., and Zhang, R. Q., "Optimal stochastic scheduling of a two-stage tandem queue with parallel servers," *Advances in Applied Probability*, vol. 31, pp. 1095–1117, Dec. 1999.

[4] Ahn, H.-S., Duenyas, I., and Zhang, R. Q., "Optimal control of a flexible server," *Advances in Applied Probability*, vol. 36, pp. 139–170, Mar. 2004.

[5] Ahn, H.-S. and Righter, R., "Dynamic load balancing with flexible workers," *Advances in Applied Probability*, vol. 38, no. 3, pp. 621–642, 2006.

[6] Andradóttir, S., Ayhan, H., and Down, D. G., "Compensating for failures with flexible servers," *Operations Research*, vol. 55, pp. 753–768, July 2007.

[7] Andradóttir, S., Ayhan, H., and Down, D. G., "Queueing systems with synergistic servers," *Operations Research*, vol. 50, pp. 772–780, 2011.

[8] Andradóttir, S., Ayhan, H., and Down, D. G., "Optimal assignment of servers to tasks when collaboration is inefficient," *Queueing Systems*, vol. 75, pp. 79–110, 2013.

[9] Andradóttir, S. and Ayhan, H., "Throughput maximization for tandem lines with two stations and flexible servers," *Operations Research*, vol. 53, pp. 516–531, May 2005.

[10] Andradóttir, S., Ayhan, H., and Down, D. G., "Design principles for flexible systems," *Production and Operations Management*, vol. 22, pp. 1144–1156, 2013.

[11] Andradóttir, S., Ayhan, H., and Down, D. G., "Server assignment policies for maximizing the steady-state throughput of finite queueing systems," *Management Science*, vol. 47, pp. 1421–1439, Oct. 2001.

[12] Andradóttir, S., Ayhan, H., and Down, D. G., "Dynamic server allocation for queueing networks with flexible servers," *Operations Research*, vol. 51, pp. 952–968, Nov. 2003.

[13] Andradóttir, S., Ayhan, H., and Down, D. G., "Dynamic assignment of dedicated and flexible servers in tandem lines," *Probability in the Engineering and Informational Sciences*, vol. 21, pp. 497–538, Oct. 2007.

[14] Andradóttir, S., Ayhan, H., and Kirkizlar, E., "Flexible servers in tandem lines with setup costs," *Queueing Systems*, vol. 70, pp. 165–186, Oct. 2011.

[15] Argon, N. T. and Tsai, Y.-C., "Dynamic control of a flexible server in an assembly-type queue with setup costs," *Queueing Systems*, vol. 70, pp. 233–268, 2012.

[16] Argon, N. T. and Andradóttir, S., "Partial pooling in tandem lines with cooperation and blocking," *Queueing Systems*, vol. 52, pp. 5–30, Jan. 2006.

[17] Arumugam, R., Mayorga, M. E., and Taaffe, K. M., "Inventory based allocation policies for flexible servers in serial systems," *Annals of Operations Research*, vol. 172, pp. 1–23, Nov. 2008.

[18] Ayesta, O., Boxma, O. J., and Verloop, I. M., "Sojourn times in a processor sharing queue with multiple vacations," *Queueing Systems*, vol. 71, pp. 53–78, 2012.

[19] Bartholdi, J. and Eisenstein, D., "A production line that balances itself," *Operations Research*, vol. 44, no. 1, pp. 21–34, 1996.

[20] Batta, R., Berman, O., and Wang, Q., "Balancing staffing and switching costs in a service center with flexible servers," *European Journal of Operational Research*, vol. 177, no. 2, pp. 924–938, 2007.

[21] Bertsimas, D. and Tsitsiklis, J. N., *Introduction to Linear Programming*. Athena Scientific, 1997.

[22] Bischak, D., "Performance of a manufacturing module with moving workers," *IEE Transactions*, vol. 28, pp. 723–733, 1995.

[23] Borst, S. C., Boxma, O. J., Morrison, J. A., and Núñez Queija, R., "The equivalence between processor sharing and service in random order," *Operations Research Letters*, vol. 31, pp. 254–262, 2003.

[24] Brandt, A. and Brandt, M., "Waiting times for M/M/1 systems under state-dependent processor sharing," *Queueing Systems*, vol. 59, pp. 297–319, 2008.

[25] Brown, G. G., Geoffrionj, A. M., and Gordon, H., "Production and sales planning with limited shared tooling at the key operation," *Management Science*, vol. 27, no. 3, pp. 247–259, 1981.

[26] Chen, N. and Jordan, S., "Throughput in processor sharing queues," *IEEE Transactions on Automatic Control*, vol. 52, no. 2, pp. 299–305, 2007.

[27] Coffman, E. G. and Kleinrock, L., "Feedback queueing models for time-shared systems," *Journal of the Association for Computing Machinery*, vol. 15, no. 4, pp. 549–576, 1968.

[28] Davis, W. J. and Flanders, S. W., "Scheduling a flexible manufacturing system with tooling constraints: An actual case study," *Interfaces*, vol. 25, no. 2, pp. 42–54, 1995.

[29] Duenyas, I., Gupta, D., and Olsen, T. L., "Control of a single server tandem queueing system with setups," *Operations Research*, vol. 46, pp. 218–230, 1998.

[30] Duenyas, I. and Van Oyen, M. P., "Stochastic scheduling of parallel queues with set-up costs," *Queueing Systems*, vol. 19, pp. 421–444, 1995.

[31] Farrar, T., "Optimal use of an extra server in a two station tandem queueing network," *IEE Transactions*, vol. 38, no. 8, pp. 1296–1299, 1994.

[32] Fayolle, G., Mitrani, I., and Iasnogorodski, R., "Sharing a processor among many job classes," *Journal of the Association for Computing Machinery*, vol. 27, no. 3, pp. 519–532, 1980.

[33] Gargeya, V. B. and Deane, R. H., "Scheduling in the dynamic job shop under auxiliary resource constraints: A simulation study," *International Journal of Production Research*, vol. 37, pp. 2817–2834, 1999.

[34] Glazebrook, K. D., "On stochastic scheduling with precedence relations and switching costs ," *Journal of Applied Probability*, vol. 17, no. 4, pp. 1016–1024, 2012.

[35] Gurumurthi, S. and Benjaafar, S., "Modeling and analysis of flexible queueing systems," *Naval Research Logistics*, vol. 51, pp. 755–782, Aug. 2004.

[36] Hajek, B., "Optimal control of two interacting service stations," *IEE Transactions*, vol. 29, no. 6, pp. 491–499, 1984.

[37] Hasenbein, J. J. and Kim, B., "Throughput maximization for two station tandem systems: A proof of the Andradóttir-Ayhan conjecture," *Queueing Systems*, vol. 67, pp. 365–386, Apr. 2011.

[38] Hillier, F. S. and Boling, R. W., "The effect of some design factors on the efficiency of production lines with variable operation times," *The Journal of Industrial Engineering*, vol. 17, no. 1, pp. 651–657, 1966.

[39] Hopp, W. J., Tekin, E., and Van Oyen, M. P., "Benefits of skill chaining in serial production lines with cross-trained workers," *Management Science*, vol. 50, no. 1, pp. 83–98, 2004.

[40] Hopp, W. and Oyen, M., "Agile workforce evaluation: A framework for cross-training and coordination," *IIE Transactions*, vol. 36, pp. 919–940, Oct. 2004.

[41] Iravani, S. M. R., Posner, M. J. M., and Buzacott, J. A., "A two-stage tandem queue attended by a moving server with holding and switching costs," *Queueing Systems*, vol. 26, pp. 203–228, 1997.

[42] Jordan, W. and Graves, S., "Principles on the process benefits of manufacturing flexibility," *Management Science*, vol. 41, no. 4, pp. 577–594, 1995.

[43] Kim, H.-W., Yu, J.-M., Kim, J.-S., Doh, H.-H., Lee, D.-H., and Nam, S.-H., "Loading algorithms for flexible manufacturing systems with partially

grouped unrelated machines and additional tooling constraints," *The International Journal of Advanced Manufacturing Technology*, vol. 58, pp. 683–691, June 2011.

[44] Kirkizlar, E., Andradóttir, S., and Ayhan, H., "Profit maximization in flexible serial queueing networks," *Queueing Systems*, vol. 77, no. 4, pp. 427–464, 2014.

[45] Kleinrock, L., "Time shared systems: A theoretical treatment," *Journal of the Association for Computing Machinery*, vol. 14, no. 2, pp. 242–261, 1967.

[46] Kleinrock, L. and Muntz, R. R., "Optimal control of service in tandem queues," *Journal of the Association for Computing Machinery*, vol. 19, no. 3, pp. 464–482, 1972.

[47] Kleinrock, L. and Nilsson, A., "On optimal scheduling algorithms for time-shared systems," *Journal of the Association for Computing Machinery*, vol. 28, no. 3, pp. 477–486, 1981.

[48] Koole, G., "Assigning a single server to inhomogeneous queues with switching costs," *Theoretical Computer Science*, vol. 182, no. 1-2, pp. 203–216, 1997.

[49] MacGregor Smith, J., "System capacity and performance modeling of finite buffer queueing networks," *International Journal of Production Research*, vol. 52, pp. 3125–3163, 2014.

[50] MacGregor Smith, J., "Queue decomposition and finite closed queueing network models," *Computers and Operations Research*, vol. 53, pp. 176–193, 2015.

[51] Mandelbaum, A. and Stolyar, A. L., "Scheduling flexible servers with convex delay costs: Heavy-traffic optimality of the generalized cμ-rule," *Operations Research*, vol. 52, pp. 836–855, Nov. 2004.

[52] Mayorga, M. E., Taaffe, K. M., and Arumugam, R., "Allocating flexible servers in serial systems with switching costs," *Annals of Operations Research*, vol. 172, no. 1, pp. 231–242, 2009.

[53] Mayorga, M. E., Taaffe, K. M., and Arumugam, R., "Allocating flexible servers in serial systems with switching costs," *Annals of Operations Research*, vol. 172, pp. 231–242, May 2009.

[54] Pandelis, D. G., "Optimal control of flexible servers in two tandem queues with operating costs," *Probability in the Engineering and Informational Sciences*, vol. 22, pp. 107–131, Dec. 2007.

[55] Pestien, V. and Ramakrishnan, S., "Asymptotic behavior of large discrete-time cyclic queueing networks," *The Annals of Applied Probability*, vol. 4, no. 2, pp. 591–606, 1994.

[56] Puterman, M. L., *Markov Decision Processes*. John Wiley & Sons, New York, NY, 1994.

[57] Reiman, M. I. and Wein, L. M., "Dynamic scheduling of a two-class queue with setups," *Operations Research*, vol. 46, no. 4, pp. 532–547, 1998.

[58] Resing, J. A. C., Hoogiehemstra, G., and Keane, M. S., "The M/G/1 processor sharing queue as the almost sure limit of feedback queues," *Journal of Applied Probability*, vol. 27, no. 4, pp. 913–918, 1990.

[59] Rosberg, Z., Varaiya, P., and Walrand, J., "Optimal control of service in tandem queues," *IEE Transactions*, vol. 27, no. 3, pp. 600–610, 1982.

[60] Schiefermayr, K. and Weichbold, J., "A complete solution for the optimal stochastic scheduling of a two-stage tandem queue with two flexible servers," *Journal of Applied Probability*, vol. 42, no. 3, pp. 778–796, 2005.

[61] Sennott, L., Van Oyen, M. P., and Iravani, S., "Optimal dynamic assignment of a flexible worker on an open production line with specialists," *European Journal of Operational Research*, vol. 170, no. 2, pp. 541–566, 2006.

[62] Shanthikumar, G. J. and Yao, D. D., "Optimal server allocation in a system of multi-server stations," *Management Science*, vol. 33, no. 9, pp. 1173–1180, 1987.

[63] Sheikzadeh, M., Benjaafar, S., and Gupta, D., "Machine sharing in manufacturing systems: Total flexibility versus chaining," *International Journal of Flexible Manufacturing Systems*, vol. 10, no. 4, pp. 351–378, 1998.

[64] Tekin, S., Andradóttir, S., and Down, D. G., "Dynamic server allocation for unstable queueing networks with flexible servers," *Queueing Systems*, vol. 70, pp. 45–79, Sept. 2011.

[65] Tsai, Y. and Argon, N., "Dynamic server assignment policies for assembly-type queues with flexible servers," *Naval Research Logistics*, vol. 55, no. 3, pp. 234–251, 2008.

[66] Van Oyen, M. P., Gel, E., and Hopp, W. J., "Performance opportunity for workforce agility in collaborative and noncollaborative work systems," *IIE Transactions*, vol. 33, no. 9, pp. 761–777, 2001.

[67] Wu, C., Lewis, M., and Veatch, M., "Dynamic allocation of reconfigurable resources in reliability considerations," *IEE Transactions*, vol. 51, no. 2, pp. 309–314, 2006.

[68] Zhuang, L. and Hindi, K. S., "Approximate decompostion for closed queueing network models of FMSs with a block-and-wait and state-dependent routing mechanism," *European Journal of Operational Research*, vol. 67, pp. 373–386, 2003.